IUTAM Bookseries

Karin Nachbagauer
Alexander Held *Editors*

IUTAM Optimal Design and Control of Multibody Systems

Proceedings of the IUTAM Symposium

Springer

IUTAM Bookseries **42**

Karin Nachbagauer · Alexander Held
Editors

Optimal Design and Control of Multibody Systems

Proceedings of the IUTAM Symposium

Springer

Editors
Karin Nachbagauer
University of Applied Sciences Upper Austria
Wels, Austria

Technical University of Munich, Institute for Advanced Studies TUM–IAS
Garching, Germany

Alexander Held
Institute of Mechanics and Ocean Engineering
Hamburg University of Technology
Hamburg, Germany

ISSN 1875-3507 ISSN 1875-3493 (electronic)
IUTAM Bookseries
ISBN 978-3-031-49999-9 ISBN 978-3-031-50000-8 (eBook)
https://doi.org/10.1007/978-3-031-50000-8

Preface

In recent years, many researchers have addressed the optimal structural design and control design of rigid and flexible multibody systems. Using the adjoint approach it is, for instance, possible to perform large-scale topology optimizations to find body designs that are optimal with respect to the time- and design-dependent loads. However, there remain many open questions and challenges for the efficient modeling, analysis, optimization, and control, also with the focus for real-time applications or artificial intelligence aspects.

The aim of the IUTAM Symposium on Optimal Design and Control of Multibody Systems 2022 was to bring together researchers working on various aspects of the sensitivity analysis, optimization, and optimal control of multibody systems. This interdisciplinary nature of the symposium allowed for the exchange of valuable information and led to fruitful discussions about optimal structural design and control design. The symposium provided a forum for discussions on relevant research issues and served as a meeting point for international researchers, scientists, and experts from academia and research institutions. The hospitality of the city of Hamburg has contributed to the success of the symposium by providing a culturally interesting, modern, innovative, yet relaxing urban environment.

We have cordially invited researchers to the IUTAM Symposium 2022 to present their work related to recent developments and novel approaches in

- (adjoint) sensitivity analysis,
- structural optimization,
- optimal control,
- robust optimization,
- artificial intelligence and machine learning,
- computational methods and software development,

for the analysis and optimization of problems from structural dynamics, rigid/flexible multibody systems, and general nonlinear dynamics.

We had 44 participants in total with 35 presentations divided into the two main areas Optimal Control and Optimal Design in the program.

We invited for contributions to a full paper version for the present IUTAM book, resulting in 14 peer-reviewed proceeding papers, 7 each from the Optimal Design and Optimal Control panels.

Acknowledgement

We are grateful to all participants for their high-quality contributions in building the basis for the symposium's success.

Moreover, we would like to thank Wolfgang Steiner, Dieter Bestle, Timo Schmidt, Jorge Ambrosio, Ali Azari Nejat, Pawel Malczyk, Ronald Aarts, and Katrin Ellermann very much for their support as Session Chairs.

We are also very much indebted to the members of the Scientific Committee

- Oliver Brüls (University of Liege, Belgium),
- Alexander Held (Hamburg University of Technology, Germany, Chairman),
- Makoto Iwamura (Kyushu University, Japan),
- Karin Nachbagauer (University of Applied Sciences Upper Austria, Austria, Chairman),
- Corina Sandu (Virginia Polytechnic Institute and State University, USA),
- Radu Serban (University of Wisconsin-Madison, USA),
- Qirong Tang (Tongji University, China),

and, in particular, the IUTAM Representative

- Peter Eberhard (University of Stuttgart, Germany)

for their valuable suggestions and advice in organizing the symposium considering the special character of IUTAM Symposia.

For the financial and organizational support, our thanks go to IUTAM, the German Research Foundation (DFG), and the Hamburg University of Technology.

At the Hamburg University of Technology, we thank Robert Seifried, Head of the Hosting Institute of Mechanics and Ocean Engineering, for supporting the symposium from the first hour. The personal involvement of the collaborators and colleagues of the Institute of Mechanics and Ocean Engineering who contributed to the organization is very much appreciated. Special thanks go to the student assistants and scientific assistants Rana Ahmed, Svenja Drücker, Helge Grossert, Malte Grube, David Schöbel, and Tobias Rückwald, for actively supporting the conference conduct.

Many thanks also to the hosts of the social events, especially for the exciting afternoon at AIRBUS, the welcome reception at the Alster boat tour, and the conference dinner at the Hotel Hamburg Hafen with the fantastic view over the city and the harbor. Most symposium participants participated in these social activities, making the symposium a successful event apart from the scientific program.

June 2023

Alexander Held
Karin Nachbagauer

Contents

Optimal Design

Optimization Processes for Automated Design of Industrial Systems

Dieter Bestle(✉)

Engineering Mechanics and Vehicle Dynamics, Brandenburg University of Technology, 03046 Cottbus, Germany
bestle@b-tu.de

Abstract. Mechanics was one of the earliest application fields demonstrating the power of rule-based deduction of general formalisms from rather few basic axioms. The usual transformation of design problems into classical optimization problems and their solution by nonlinear programming algorithms follows the same principle, and is thus mainly used by the mechanics and control community. System design on an industrial scale, however, is a much more challenging creative task, which cannot be formalized so easily, which is why it is mostly still human-driven. In order to break up this game stopper, algorithms for multi-criterion optimization, function approximation and statistical sensitivity analysis may be integrated in a common design process. Especially the emerging field of data-driven artificial intelligence (AI) methods may become the pushing game changer. The paper will demonstrate major challenges of industrial design and some solution strategies to bridge the gap between engineering design intuition and formalized problem solution.

1 Conventional System Design

System design is generally searching for best problem solutions. On an industrial scale, this is often achieved based on human intelligence, experience and intensive parameter studies. However, such a practise is not only rather ineffective, but unsuitable to exploit the full optimization potential. On the one hand, humans can imagine effects of design changes only in one- or low-dimensional design spaces, where it is proven that subsequent optimization in subspaces doesn't lead to the overall optimum. On the other hand, engineers can be real experts only in a few disciplines, whereas system design is multi-disciplinary involving multiple departments, where subsequent involvement in design decisions leads to the same difficulties of not exploiting the full optimization potential.

Numerical optimization has a much higher potential to find best solutions in high-dimensional design spaces. The classical formulation of an optimization problem is

$$\min_{\mathbf{p}\in P} f(\mathbf{p}) \ \ where\ \ P = \left\{ \mathbf{p} \in \mathbb{R}^D \middle| \mathbf{g}(\mathbf{p}) = \mathbf{0}, \mathbf{h}(\mathbf{p}) \le \mathbf{0}, \mathbf{p}^l \le \mathbf{p} \le \mathbf{p}^u \right\}. \tag{1}$$

The D-dimensional design space P is typically spanned by variable model parameters acting as design variables, such as geometric dimensions, mass geometry, stiffness and damping of multibody systems (e.g. [1]), control parameters of mechatronic systems

https://doi.org/10.1007/978-3-031-50000-8_1

(e.g. [2]), or structural properties of elastic systems (e.g. [3]). This design space may be limited by equality or inequality constraints as well as bounds resulting from physical limits or design restrictions. Design objectives $\tilde{f}(\mathbf{p})$ to be maximized can be easily transformed into their negative counterparts $f(\mathbf{p}) = -\tilde{f}(\mathbf{p})$ to be minimized.

Especially the 1960s have been a golden age of developing powerful solution algorithms [4] for solving problem (1), mostly based on gradients which may be provided purely numerically by divided differences, semi-analytically by direct differentiation or as adjoints utilizing computer algebra (e.g. [1]), or fully automatically by differentiating computer code either generating new code or using operation overloading (e.g. [5]). Only in few disciplines, especially structural design, these algorithms have become an inherent part of finite element codes for natural use on an industrial scale [6]. For general engineering design problems, however, the expectation that from then all design problems may be solved rather easily by optimization was mostly disappointed by the fact that the solutions enforced by scalar optimization are often totally impractical.

2 Why Classical Optimization Cannot Solve Industrial Design Problems

One of the main reasons for failure is that in most cases there doesn't exist any single optimal solution, but practical design decisions are always tradeoffs between conflicting demands and desires. Further, industrial design tasks are characterized by the following features:

i. design intentions are often intuitive, which can hardly be formalized;
ii. design parametrization is often analysis-oriented being less qualified for optimization;
iii. the design space is often high-dimensional and heterogenous including continuous and discrete parameters as well as shape functions;
iv. design demands are typically multi-disciplinary enforcing various analysis software including CAD, CFD, MBS, FEM, control, as well as design rules based on expertise from different departments;
v. often design analysis is performed with commercial-of-the-shelf (COTS) software resulting in costly, noisy black-box results with limited robustness;
vi. sometimes software runs on heterogenous hardware with different operation systems;
vii. typically multi-modal, multi-fidelity design criteria with unclear correlation are to be expected;
viii. deterministic optimization of nominal designs is often not sufficient due to manufacturing uncertainties or uncertain operation conditions.

This property list is concluded from observations within several decades of industrial cooperation and may be motivated by some examples. The first example regards the design of a new steering concept solely based on differences in the drive forces F_l, F_r exerted by electric in-wheel motors, which is called differential steering [7], Fig. 1a. The basic control concept uses a virtual reference car to generate reference values for sideslip angle β and yaw rate ω. A LQR-controller (linear-quadratic regulator) then tries to follow these reference values with appropriate wheel torques turning the wheel direction by differences between left and right drive forces as part of feedback control vector **u**. The point here is that, although LQR design minimizing the state error **x** and control effort is already a formalized optimization problem with a clearly defined objective $J = \int(\mathbf{x}^T\mathbf{R}\mathbf{x} + \mathbf{u}^T\mathbf{Q}\mathbf{u})\,\mathrm{d}t$ to be minimized and a well-known solution, the selection of **Q**- and **R**-matrices is totally artificial and may generate any result far from desired vehicle behavior. Automotive design engineers, therefore, use a totally different concept, where assessment is based on visible results from specific road tests like driving on a circle with increasing speed v to obtain over- or under-steering characteristics and stability limits, or step-changes on steering wheel to assess agility, or a double-lane change maneuver for the same reasons. This conflict may be resolved by a two-step approach where matrices **Q** and **R** are considered as part of the design vector **p** and optimized together with mechanical model parameters based on time-consuming simulations of exactly the same road test maneuvers [7].

A rather similar example is the design of a rear wheel steering device based on a linear vehicle model [8]. The design intention here is based on frequency response curves for lateral acceleration $\ddot{y}$ and yaw rate $\dot{\psi}$ expressing stability and agility with some intuitive desire. One point here is the speed dependence enforcing functional relations for the control gains as design quantities, the second one is the multiplicity of amplitude response curves $a_{\ddot{y}}$ and $a_{\dot{\psi}}$ shown in Fig. 1b due to uncertainties of car loading and tire stiffness. Optimization here has to account first for smooth, low-dimensional design parametrization and second for the high effort of statistically assessing probabilistic driving criteria to make the final design being robust.

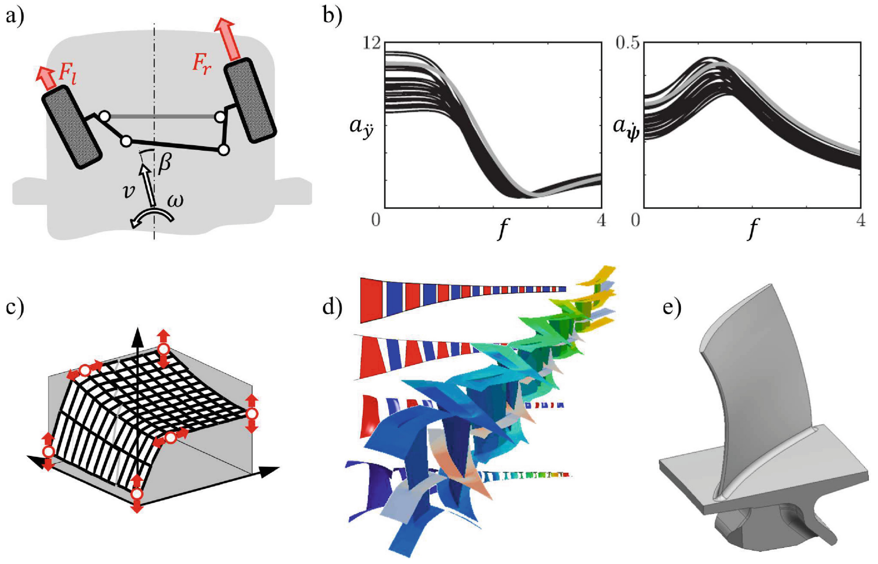


Fig. 1. Design examples: a) differential steering; b) frequency response for lateral vehicle dynamics; c) automatic gear control; d) multi-fidelity compressor design; e) blade design.

The next example regards the calibration of automatic gears which in industrial practice is done on public roads by experienced calibration engineers directly changing some parameters of the gear control unit and assessing launch or gear shift quality by subjective driving impressions. Typical launch criteria, which can also be measured in real tests, are, e.g., sportiness expressed by the required time for speeding up within a pre-defined distance and the corresponding end velocity, alternatively the synchronization time between car and motor speed, further comfort expressed by minimal jerk, or clutch wear regarded by friction dissipation energy. They all have been developed over many years of experience, but in the formal approach of [9] it turned out that they are highly correlated which disturbs optimization due to the unnecessarily high dimension of the criterion space. Also the design space dimension is unnecessarily high because in practice control characteristics are described by point clouds instead of functional relations [10], Fig. 1c. Finally robustness against changing friction coefficients, oil temperature or clutch stiffness due to plate wear is an issue increasing the simulation effort dramatically [11].

As an example for multi-disciplinary design with heterogeneous design variables, the design of automotive shock absorbers may be mentioned [12]. In practice, the internal valve design determining the frequency- and amplitude-dependent stroke-force characteristic of an hydraulic damper is performed by experienced engineers combining various shims from an assortment box, building up the test damper, applying it to a specific car to be designed, and then assessing the resulting driving behavior in time-consuming field tests. In order to find proper shim stacks automatically, computational fluid dynamics (CFD) is required to obtain the pressure distribution within the valve channels and thus onto the shim stacks, and the finite element method (FEM) is required to obtain the corresponding shim pressure-deflection behavior which then effects the oil flow determining the stroke velocity. Both black-box programs have to be integrated into a common design loop to solve the combinatorial problem of stacking shims with partly continuous, but mostly discrete values for thicknesses and diameters. Additionally uncertainties due to friction between shims and manufacturing imperfections of thickness and diameter have to be taken into account.

As last, most challenging example, aero engine design may be considered. Due to its complexity it is typically split into several design subtasks for fan, compressor, combustor and turbine. However, even the compressor design is already so challenging that several models of various fidelity have to be combined reaching from 1D-flow computations via 2D-CFD in axial-radial and axial-circumferential directions till full 3D-CFD to make various design decisions [13], Fig. 1d. After fixing major flow characteristics, the next step is to design proper blade shapes which account for both aerodynamic aspects as well as structural aspects requiring a combination of computer-aided design (CAD) for geometry definition, CFD, FEM and design rules regarding resonance and flutter [14]. Especially CFD requires smooth blade parametrization to allow flow calculations (Fig. 1e). Since even minor manufacturing deviations have tremendous impact on engine efficiency, robust design is inevitable [15]. In order to exploit the full optimization potential, todays industrial design strategy, which typically fixes component requirements based on simple thermomechanical models and then provides these specifications to the different component departments as design challenges, is no longer sufficient. Instead, holistic engine design based on process integration of the best available analysis models and cooperative optimization is required.

3 Strategies for Meeting Industrial Design Challenges

The above collection of examples should have clearly enough shown that industrial design challenges are so manifold that the academic optimization approach (1) cannot master them, but has to be accomplished by some additional design strategies to be proposed in the following.

3.1 Gradient Avoidance Through Population-Based Optimization

Although gradient-based algorithms like sequential quadratic programming (SQP) are rather efficient for solving optimization problem (1), they are not the first choice for industrial design problems for many reasons. Analytical computation of gradients with computer algebra is reserved for academic toy problems. Providing semi-analytical equations for computing gradients takes too much preparation time and requires rather specific formulations of the underlying analysis model, see e.g. [1] for multibody dynamics. Automatic differentiation is also no option when using COTS programs offering adjoint codes only in very rare cases. If automatic meshing in CFD and FEM is used, numerical differentiation with divided differences is also no alternative since mesh perturbations yield noisy analysis results prohibiting use of small design perturbations required for obtaining precise results. In some cases, gradients may even not exist due to non-smooth criterion functions. Finally, the downhill property of gradient-based optimization algorithms prohibits the finding of the desired global optimum in case of multi-modal functions with several local optima, which has to be expected in industrial design applications at any time.

Thus, global optimization algorithms like genetic algorithms (GA [16]), particle swarm optimization (PSO [17]) and differential evolution (DE [18]) are often the better choice. They are directly applicable to any problem allowing to provide analysis results, do not need any preparation time for gradient computation and find very good or even multiple good solutions with high probability. The reason is that they gain a bigger picture of the design space by iteratively developing complete sets of multiple, widely spread designs, so called design populations, instead of improving just a single design point. This, however, requires already a good first overview on the design space by properly initializing the first population of design individuals, e.g. by Latin hypercube sampling [19].

Major drawbacks of all global optimization algorithms are the extremely high number of required design evaluations and that constraints can be handled by penalty strategies only. The first aspect will be addressed in Sect. 3.3, the latter is less challenging with gradient-free algorithms than with effective nonlinear programming algorithms requiring smooth penalty functions. For example, in case of a single inequality constraint $h(\mathbf{p}) \leq 0$, adding a constant term to a quadratic penalty results in the non-smooth penalty function

$$\sigma(\mathbf{p}) = \begin{cases} \sigma_0 + h^2(\mathbf{p}) & if\ h(\mathbf{p}) > 0, \\ 0 & else. \end{cases} \tag{2}$$

By choosing $\sigma_0 > 0$ big enough, minimization of the penalized objective $f(\mathbf{p}) + \sigma(\mathbf{p})$ will finally fulfill the constraint almost exactly. Above optimization procedures may even be applied to non-robust function evaluation, if analysis failure, e.g. due to meshing problems in CFD or FEM, is treated properly by returning a penalized function value, because in evolutionary optimization algorithms penalized designs will be eliminated by the subsequent selection step anyway. Since these kinds of population-based optimization algorithms are able to learn from actual design data, they are considered as AI-strategies.

3.2 More Practicability Through Multi-objective Optimization

Industrial system design typically involves several departments from several disciplines, each with its own design desires and demands which hardly can be summarized in a single objective. Thus, the scalar optimization problem (1) has to be reformulated as

$$\min_{\mathbf{p}\in P} \mathbf{f}(\mathbf{p}) \textit{ where } P = \left\{ \mathbf{p} \in \mathbb{R}^D \middle| \mathbf{g}(\mathbf{p}) = \mathbf{0}, \mathbf{h}(\mathbf{p}) \leq \mathbf{0}, \mathbf{p}^l \leq \mathbf{p} \leq \mathbf{p}^u \right\} \tag{3}$$

where now several objectives $f_i(\mathbf{p})$, which shall be minimized simultaneously, are summarized in the vector criterion $\mathbf{f}(\mathbf{p}) = [f_1, f_2, ..., f_n]^T$. The ideal solution, however, where all objectives take their minimum value, exists only in the rare case of highly correlated criteria. Usually the objectives are conflicting in the sense that improvement of one objective will worsen at least another one. In this case, multi-objective optimization delivers a strict mathematical concept of defining and finding tradeoffs, so called Pareto optima [20], from which the designer may choose anyone according to additional preferences.

Moreover, multi-objective extensions of GA and PSO, named MOGA (e.g. [20]) and MOPSO (e.g. [21]), provide easy-to-use algorithms to find non-dominated solutions as a representative set of Pareto optima in a single run without any user-interaction or a priori knowledge. These codes follow the same evolutionary principles as their single-objective counterparts, only the selection step has to be modified. Typically a rank-sorting selection criterion enforces convergence towards the Pareto front, whereas some distance criterion enforces spreading of designs over the whole front to provide a representative full picture of possible tradeoffs to the design engineer. Thus, multi-objective optimization releases the design engineer from the search of optimal tradeoffs without depriving him of the last design decision.

3.3 More Efficiency Through Metamodeling

Population-based multi-objective optimization algorithms have the same drawbacks as their scalar counterparts mentioned in Sect. 3.1; especially the very high number of required design evaluations conflicts with the high time-consumption of industrial analysis tools. This seems to disqualify such algorithms for industrial application where each single design assessment may take minutes to hours of computation time, especially if structural or flow analysis is involved. However, this contradiction can be resolved in various ways. First, multi-objective optimization allows fully parallel evaluation of the involved objectives and constraint functions on high-performance computer clusters; second, population-based optimization allows fully parallel evaluation of the individuals of the traced population. Both, however, require multiple licenses, typically one for each cluster node, if COTS software is used.

A better concept is to insert a metamodel layer partly decoupling optimization from analysis. Metamodels, also called surrogates or response functions, are analytical functions approximating the input-output relations $f_i(\mathbf{p})$ of objectives or constraint functions. They can be evaluated within fractions of seconds and thus serve as computationally cheap assessment source interacting with population-based optimization algorithms. Well-established surrogate examples are Kriging [22] and radial-basis functions [23], less qualified are e.g. polynomials due to their strict correlation between function complexity and the required number of support points. High potential have supervised-learning AI-strategies like regression trees of artificial neural networks (e.g. [7]) as universal function generators. In contrast to numerical regression, where interpolation of a given data set would be considered best, AI focuses not so much on the quality of representing the given data set, but on good generalization properties of the resulting metamodel w.r.t. any future data not seen during training. Thus, interpolation would be considered as overfitting, whereas model training in AI is stopped if the generalization error on new data is lowest.

Typically metamodels are trained only once and then used as basis of optimization. This however, is not recommended since it requires equal approximation quality in the whole design space which wastes a lot of costly function evaluations in non-interesting design regions and is hard to get in higher-dimensional design spaces. A better concept is to train an initial metamodel with only a small data set of analyzed support points and then perform a first optimization for getting inside where potential optima may be located. In such regions additional designs may be selected and analyzed with the original time-consuming analysis codes to re-train and improve the metamodel in the interesting areas. Subsequent optimization and iterative refinement of the metamodel may be integrated into a common algorithm [24]. The criteria for selecting new support points can be similar to the selection step of evolutionary algorithms, but support point concentration should be avoided, first because of waste of computing resources and second because some surrogates may suffer from singularity problems. Additionally the approximation uncertainty being e.g. available from Kriging models may be included. For scalar optimization problems, expected improvement is an advanced algorithm balancing the exploration of design regions with uncertain approximation and exploiting regions with already found very good designs [25].

The master-slave approach in Fig. 2 allows a fully decoupled, totally parallel multi-objective optimization involving models of different fidelity and analysis time [26]. An initial data set generated with design-of-experiments (e.g. Latin hypercube sampling) is written to a working space where various slave processes may pick up designs to be evaluated according to specific functions and disciplines. A second group of slave-processes is awaiting the results to train or re-train response surface models (RSM). The master optimization process is only working on these metamodels and proposes new designs as support points which enhance the design set in the working space. Due to decoupled analysis and training, fast and simple analyses are able to generate high-quality surrogates based on many original design evaluations and drive the optimization process, whereas slow high-fidelity analyses are only able to provide rough metamodels from few costly design evaluations, but prohibit optimization from getting lost in poor design regions.

3.4 Constraint Avoidance Through Optimization-oriented Parametrization

Most optimization algorithms presume real-valued design variables like in optimization problems (1) or (3). This is the case in many applications, where any specific model parameter may serve as design variable. However, model parametrization is often analysis-oriented resulting in many restrictions and dependences to be considered to guarantee a feasible design. Human designers would account for informal geometric restrictions automatically during search, whereas probabilistic search algorithms like GA require strict formalized constraint equations which may be hard to be fulfilled as discussed in Sect. 3.1. Therefore, proper optimization-oriented design parametrization is a key-driver for receiving satisfying optimization results.

This is especially the case when smoothness in shape optimization problems like blade design in Fig. 1e has to be enforced to allow for flow analysis and avoid unrealistic stresses at artificial kinks of segmented parts in structural analysis. This can be achieved easily by using e.g. B-spline curves and surfaces [27] superposing basis function with freely movable control points which may partly be selected as design variables [28]. Alternatively, e.g. Hermite splines may be used where the selectable control points in Fig. 1c lie on the curve or surface, respectively, and monotonicity demands may be considered by slope restrictions [10].

In such cases, there is a free choice of the number of design parameters to be varied requiring a tradeoff between allowed shape complexity, and thus design freedom, on

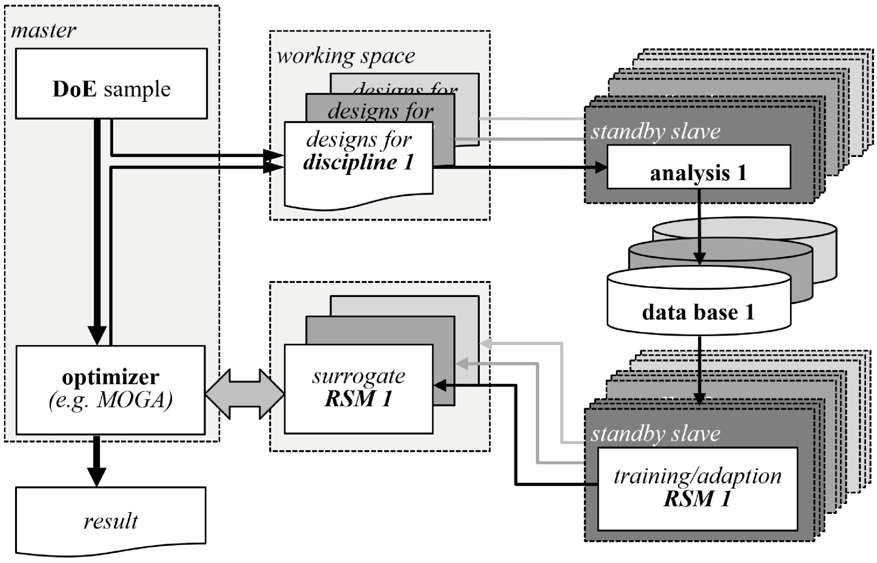


Fig. 2. Master-slave architecture for efficient global optimization.

the one hand and complexity of the optimization problem in case of too many design variables on the other hand, where especially the probabilistic search of population-based strategies suffers from the curse of dimensionality. This requires a restriction of the set of design variables on the most influential parameters in a first attempt and then sequential search in extended or changed design spaces later on.

The selection of the best design variables can only partly be made by physical reasoning; often they are too abstract for predicting their influence on objectives and constraints like in the case of shape optimization. In such cases sensitivity analysis has to be performed, where a local sensitivity analysis as used for gradient computation is not sufficient, but sensitivities have to be representative for the whole design space. The latter can be achieved e.g. with AI-methods building up different surrogates for objectives and constraints from a representative DoE, one with all possible design variables involved and one with leaving out one or more design variables, respectively. The leave-out surrogate with least mean difference to the full surrogate then tells which parameters may be neglected without losing input-output information on that specific function. By leaving out design parameters successively one-at-a-time in a systematic way and using Pearson's correlation coefficient as difference criterion, parameters can be sorted according to their influence, see [28] or [7].

3.5 Robust Design

Theoretically optimization assumes that the found optimal design can be realized as computed and drives designs to the limits at the border of the feasible design space. Practically, however, probabilistic manufacturing deviations are unavoidable and the real operation conditions of the optimized part may differ from the supposed nominal conditions, which results in degradations of design objectives or even violations of constraints resulting in failure of the product. Classical engineering practice accounts for such effects by introducing safety factors where, however, their correlation with the real uncertainties and safety margins is often rather unclear. The better strategy is to assign probability distributions to uncertain quantities, choose respective samples and evaluate objectives and constraints for these samples statistically. This results in information about product quality variation and failure probabilities [29]. Such a reliability analysis may then reveal that it would have been a better choice to select a nominal design with worse nominal objective values but less objective variation, i.e., the design is more robust, or lower failure probability, i.e., the design is more reliable.

Instead of performing reliability analysis only a posteriori for the final deterministically optimized design, the better strategy is to include it into the design loop already and change the problem formulation (2) accordingly. Instead of assessing the nominal design for nominal operating conditions directly, a sample of uncertain designs about the nominal design should be generated with the prescribed probability distributions and assessed w.r.t. the uncertain operating conditions. This can be achieved e.g. by assessing the 95-percentile to be of best possible quality and fulfill design constraints or to minimize the objectives' mean and variance of the whole sample simultaneously or to prescribe failure probabilities for the sample, see [15] or [30].

It is well known that statistical estimates improve with higher sample sizes, where it should be kept in mind that already design search with global optimization algorithms requires many thousands of evaluations of nominal design points. Thus, the computational analysis effort, when adding statistical assessments, is even higher than already discussed in Sect. 3.3 for deterministic optimization. However, by using similar metamodels as discussed in Sect. 3.3 also for locally approximating the impact of design variations onto design behavior dramatically cuts down the estimation effort and allows almost arbitrarily high sample sizes. In [8] it could be demonstrated that statistical estimates based on metamodels are of several orders of magnitude better than using their support points directly.

3.6 Integration of Expert Knowledge via Supervised Learning

Rule-based formalization of design problems has its limits where human experts intuitively deduce design quality from graphical representations without traceable rules. A good example is the classification of characteristic blade vibration modes which is necessary to protect turbine blades from flutter by appropriate optimization constraints. For simple symmetric geometries like aero engine compressor blisks (blade integrated discs), such a distinction of modes may be achieved by counting radial and circumferential nodal lines [31], for general geometries like turbine blades, however, nodal lines in Fig. 3 may crisscross so complicated that this is impossible.

In such a case, human experts may label some hundred, graphically provided examples by assigning one of up to 15 different mode labels rather easily. These labeled data may then be used to train an artificial neural network (ANN) by supervised learning, which later during optimization will provide the correct classification automatically and thousandfold without further human interaction [32]. In order to close the gap between finite-element calculations of eigenmodes with varying, automatically generated meshes and the classification-ANN with prescribed structure, i.e., fixed number of input nodes, layers and output nodes, a self-organizing or Kohonen map is used to transform the changing mesh information into structured mesh information.

The new potential of AI-methods to support design is that such a learning from data may release the human designer from artificial, rule-based formalization of design criteria. Learning from data can be much easier than causal conclusive formalization of rules from physics, and especially deep learning based on convolutional neural networks (CNN [33]) has high potential to learn any kind of correlations and abstract features even if humans do not understand the rules. So supervised learning is an elegant way of querying experts for labeling data based on their non-formal knowledge and then learning rules from these labeled data automatically which may finally be integrated in automatic design processes.

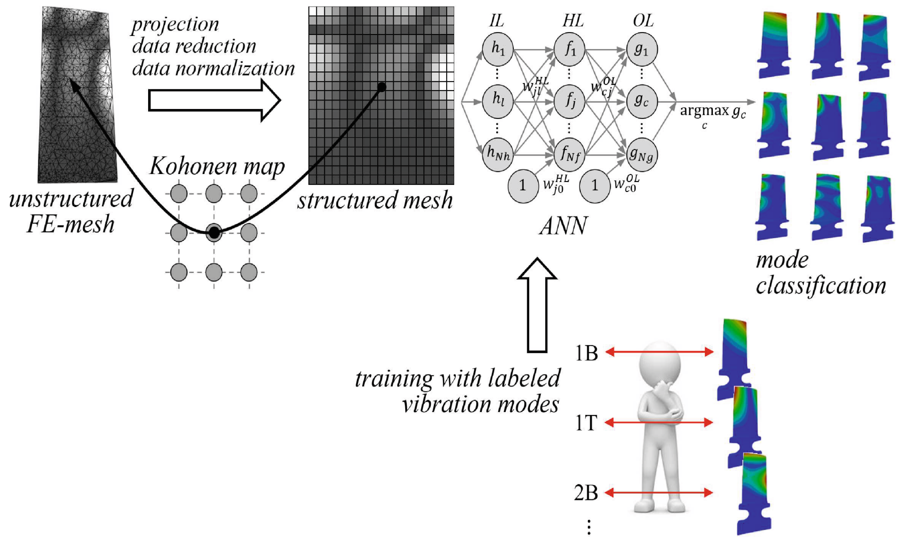


Fig. 3. Identification of vibration modes with AI [32].

4 Conclusions and Outlook

The above presentation and discussion of examples should have made clear that optimization is not restricted to academic toy problems, but mature for becoming a valuable toolset for industrial design tasks when accomplished by time-saving procedures, where beside computational time-saving the saving of preparation time for properly formalizing the optimization problem is often more important. Especially the fast-emerging AI-technologies have high potential to support and speed up system design on an industrial scale. This was the reason for establishing the priority programm SPP 2353 by the German Research Council [34] in 2022 to better link AI to design and to develop proper modules for setting up general design assistance systems.

However, AI not only changes the way systems are designed, but also the design features as it becomes part of the system to be designed. Releasing systems from strict rule-based behavior and introducing AI-components trained with data correlations is an actual paradigm change increasing system potentials dramatically by audio and vision capabilities, which so far have been reserved to the living world, but at the same time causes safety issues which have to be accounted for in the design process. Examples in [35] show how easily vision systems, which e.g. are established for autonomous driving, may be fooled by obviously insignificant changes or noise.

References

1. Bestle, D.: Analyse und Optimierung von Mehrkörpersystemen. Springer, Heidelberg (1994). https://doi.org/10.1007/978-3-642-52352-6
2. Szuster, M., Hendzel, Z.: Intelligent Optimal Adaptive Control for Mechatronic Systems. SSDC, vol. 120. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-68826-8
3. Haug, E.J., Arora, J.S.: Applied Optimal Design: Mechanical and Structural Systems. Wiley, New York (1979)
4. Fletcher, R.: Practical Methods of Optimization. Wiley, Chichester (1987)
5. Bischof, C., Lang, B., Vehreschild, A.: Automatic differentiation for MATLAB programs. Appl. Math. Mech. **2**, 50–53 (2003). https://doi.org/10.1002/pamm.200310013
6. Haftka, R.T., Sobieszczanski-Sobieski, J.: Structural optimization: history. In: Excyclopedia of Optimization, pp. 3834–3836. Springer, Boston (2008). https://doi.org/10.1007/978-0-387-74759-0_669
7. Kuslits, M., Bestle, D.: Multiobjective performance optimisation of a new differential steering concept. Veh. Syst. Dyn. **60**, 73–95 (2020)
8. Busch, J., Bestle, D.: Optimisation of lateral car dynamics taking into account parameter uncertainties. Veh. Syst. Dyn. **52**, 166–185 (2014)
9. Wehbi, K., Bestle, D., Kahlbau, S.: Multi-objective launch control optimization for automatic clutch engagement. In: Proceedings of the 6th International Conference on Experiments/Process/System Modelling/Simulation/Optimization (IC-EpsMsO), Athens (2015)
10. Wurm, A., Bestle, D., Kahlbau, S.: Automotive shift quality optimization based on piecewise monotone interpolation of parameter characteristics. In: Proceedings of the 4th International Conference on Engineering Optimization (EngOpt 2014), Lisbon (2014)
11. Wurm, A., Bestle, D.: Robust design optimization for improving automotive shift quality. J. Optimiz. Eng. **17**, 421–436 (2016)
12. Hofmann, T., Brenner, T., Bestle, D.: Investigation of the disc deflection behavior of shim valves in vehicle shock absorbers. In: Proceedings of the WCX World Congress Experience, Detroit, SAE Technical Paper 2018-01-0701 (2018)
13. Poehlmann, F., Bestle, D.: Multi-objective compressor design optimization using multi-design transfer between codes of different fidelity. In: Proceedings of the ASME Turbo Expo, Copenhagen, GT2012-68577 (2012)
14. Otto, D., Bestle, D.: Multi-disciplinary blading design by means of multi-objective optimisation. In: Proceedings of the 1st European Air and Space Conference (CEAS), Berlin (2007)
15. Flassig, P.M., Dutta, A.K., Bestle, D.: Robuste Auslegung von Verdichterschaufeln. In: Proceedings of the Deutscher Luft- und Raumfahrtkongress, Darmstadt, DLRK2008-081174 (2008)
16. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
17. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, Part IV, pp. 1942–1948 (1995)
18. Storn, R., Price, K.: Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Opt. **11**, 341–359 (1997)
19. McKay, M.D., Beckman, R.J., Conover, W.J.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **21**, 239–245 (1979)
20. Deb, K.: Multi-objective Optimization Using Evolutionary Algorithms. Wiley, Chichester (2002)
21. Coello Coello, C.A., Lechuga, M.S.: MOPSO: a proposal for multiple objective particle swarm optimization. In: Proceedings of the Congress on Evolutionary Computation (CEC 2002), pp. 1051–1056 (2002)

22. Lophaven, S.N., Nielsen, H.B., Søndergaard, J.: DACE: A MATLAB Kriging Toolbox. Techn. Report IMM-TR-2002-12. Technical University of Denmark, Lyngby (2002)
23. Buhmann, M.D.: Radial Basis Functions: Theory and Implementations. University Press, Cambridge (2004)
24. Regis, R.G., Shoemaker, C.A.: A stochastic radial basis function method for the global optimization of expensive functions. INFORMS J. Comput. **19**, 497–509 (2007)
25. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. Glob. Opt. **13**, 455–492 (1998)
26. Hartwig, L., Bestle, D.: Compressor blade design for stationary gas turbines using dimension reduced surrogate modeling. In: Proceedings of the IEEE Congress on Evoluationary Computation, Donostia, pp. 1595–1602 (2017)
27. Piegl, L., Tiller, W.: The NURBS Book. Springer, Heidelberg (1997)
28. Amtsfeld, P., Lockan, M., Bestle, D., Meyer, M.: Accelerated 3D aerodynamic optimization of gas turbine blades. In: Proceedings of the SME Turbo EXPO, Düsseldorf, GT2014-25618 (2014)
29. Bucher, C.: Computational Analysis of Randomness in Structural Mechanics. Taylor & Francis, London (2009)
30. Martin, I., Hartwig, L., Bestle, D.: A Multi-objective optimization framework for robust axial compressor airfoil design. Struct. Multidiscip. Optim. **59**, 1935–1947 (2019)
31. Beirow, B.: Grundlegende Untersuchungen zum Schwingungsverhalten von Verdichterlaufrädern in Integralbauweise, Habilitationsschrift. Shaker, Aachen (2009)
32. Martin, I., Bestle, D.: Automated eigenmode classification for airfoils in the presence of fixation uncertainties. Eng. Appl. Artif. Intell. **67**, 187–196 (2018)
33. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
34. DFG-Schwerpunktprogramm SPP 2353. Daring More Intelligence: Design Assistants in Mechanics and Dynamics, 2022–2025. https://gepris.dfg.de/gepris/projekt/460725022. Accessed 2022
35. Heaven, D.: Why deep-learning AIs are so easy to fool. Nature **574**, 163–166 (2019). https://doi.org/10.1038/d41586-019-03013-5

Optimal Lightweight Engineering Design via a Three-Block Solver Scheme for Mechanical Analysis

Erich Wehrle[1,2(✉)]

[1] Collins Aerospace, Applied Research and Technology, Munich, Germany
Erich.Wehrle@collins.com,Erich.Wehrle@unibz.it
[2] Free University of Bozen-Bolzano, Universitätsplatz 1, 39100 Bozen, South Tyrol, Italy

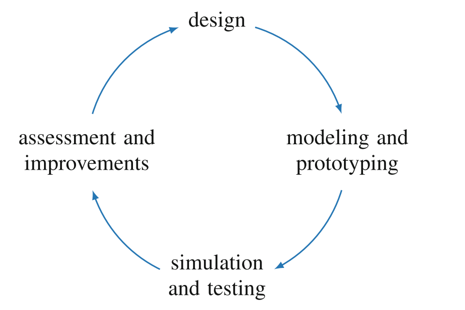
Abstract. This paper provides an introduction to structural design optimization of dynamic systems. Cast as an enabler of lightweight engineering design via its virtuous circle, design optimization is shown in collaboration with the mechanical modeling. A general form of the mechanical model is shown via a differential–algebraic equation categorized into three elementary building blocks: governing equations, nonlinear solver and time integration. The three-block solving scheme enables a modular and implementation-near view to complicated analysis. This gives a clear view to the nontrivial task of analytical sensitivity analysis, especially when using direct differentiation. This introduced methodology is applied to the gradient-based design optimization of a morphing wing using flexible multibody dynamics. The approach using analytical sensitivity analysis is compared with numerical finite differencing.

1 Design Optimization as Enabler of *Virtuous Circle of Lightweight Engineering Design*

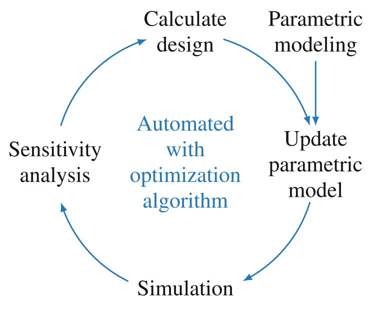
Design optimization automates the "traditional" trial-and-error iterative process of engineering design (Fig. 1). The optimization-based approach relies on mathematical algorithms to choose the designs, which are represented by parametric models that are simulated. This decreases time needed for each development cycle, leading to less expense and time for development, more exploration of the design domain and, therefore, better designs.

The use of design optimization requires the engineering design problem to be formulated as a mathematical optimization problem. This step is problem-dependent and includes choice of design variables as well as formulation of objective and constraint functions. The standard mathematical optimization problem minimizes a value so that other requirements formulated as constraints are not

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 16–29, 2024.
https://doi.org/10.1007/978-3-031-50000-8_2



(a) "Traditional" iterative engineering development



(b) Optimization-based engineering development

Fig. 1. Engineering development and design approaches

violated,

$$\begin{aligned} \min_{\underline{\mathsf{x}} \in \underline{\mathsf{X}}} \mathsf{f}\left(\underline{\mathsf{x}}\right), \qquad\qquad & (1) \\ \text{such that } \underline{\mathsf{g}}\left(\underline{\mathsf{x}}\right) \leq \underline{0}, \\ \underline{\mathsf{h}}\left(\underline{\mathsf{x}}\right) = \underline{0}, \\ \underline{\mathsf{X}} = \left[\underline{\mathsf{x}}_L \; \underline{\mathsf{x}}_U\right], \\ \text{governed by } \underline{R}\left(\underline{\mathsf{x}}\right) = \underline{0}, \end{aligned}$$

in which f is the objective function, $\underline{\mathsf{x}}$ is the vector of design variables, $\underline{\mathsf{X}}$ is the design domain defined by the upper $\underline{\mathsf{x}}_U$ and lower bounds $\underline{\mathsf{x}}_L$ of the design variables, $\underline{\mathsf{g}}$ is the vector of inequality constraint functions, $\underline{\mathsf{h}}$ the vector equality constraint functions and $\underline{R}$ is the residual of a general governing equilibrium equation, which represents the simulation. Engineering simulation possibilities are finite-element analysis, multibody simulation and computational fluid dynamics, for example. Typical design variables include geometrical, material and operational parameters. Constraints are requirements on the material, structural mechanics, dynamics and manufacturing. Objective functions can include mass, cost, or other performance measures.

Lightweight engineering design is an engineering design paradigm in which by reducing mass, structural and drive requirements can be decreased and, therefore, the structural mass can in turn be further reduced. This *virtuous circle of lightweight design* [36] (Fig. 2) is magnified with dynamic systems in which self-weight is reduced and with it, the resulting inertial forces. Structural design optimization assists engineering design to enter this virtuous circle, pushing designs to their theoretical optimum.

Despite the lineage of design optimization reaching back to [29] and multibody optimization to [2,15], there remain a number of open research questions. These include efficient sensitivity analysis, inclusion of highly nonlinear systems, mixed topology, shape and size problems, multidisciplinary and multiphysical

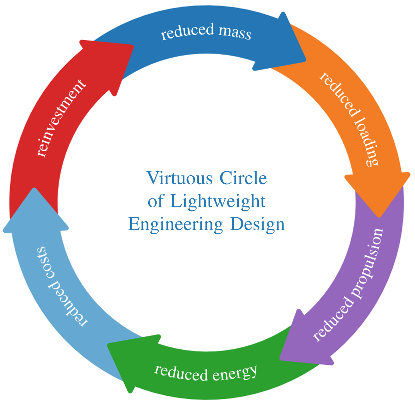


Fig. 2. Virtuous circle of lightweight engineering design

problems as well as ever-growing model size and number of design variables. Optimization of flexible multibody dynamics is reviewed in [13].

Especially analytical sensitivities can play a role to reduce computation effort, thus allowing for larger problems. Design sensitivity analysis with structural and mechanical simulation is outlined in [16,19,23,34].

The analytical sensitivities for nonlinear, time dependent problems is a non-trivial task and the three-block solver scheme put forth aims to ease this process by reusing each block for both primary and sensitivity analysis. The three-block scheme for primal analysis is introduced for the general case of differential–algebraic equations in Sect. 2. Building upon this, design sensitivity analysis using direct differentiation of the three-block scheme is shown in Sect. 3. This is followed in Sect. 4 by the application of this approach to the design optimization of a morphing wing, culminating in conclusion (Sect. 5).

2 Modeling Mechanical Systems with Differential–Algebraic Equations

Mechanical analysis includes multibody dynamics, nonlinear dynamics, crash explicit dynamics, linear dynamics, nonlinear (quasi-) statics, eigenvalue vibration, eigenvalue buckling and linear (quasi-) statics. These mechanical analyses can be generally categorized by being carried out with three elementary building blocks [36]: governing equations, nonlinear solver and time integration (see Fig. 3). The three blocks can therefore be reused for a range of analysis types.

This reduces the complexity and the effort in the nontrival task of deriving and implementing the analytical design sensitivities (Sect. 3). Although automatic differentiation methods exist [22], the manual differentiation is preferred by the author. Coming at the cost of high computational effort and being prone to human error, manual differentiation provides transparency and plausibility owing to its physics-based nature, in addition to being highly reusable and efficient. This general framework for governing equations is applied to flexible multibody dynamics.

Flexible multibody dynamics can be modeled with a system of differential–algebraic equations, comprised of a set of differential equations for the equation of motion and a set of algebraic equations for the kinematic constraints. The internal forces are calculated via the material response in the form of the constitutive equation via stresses and strains. In its simplest form the material law is isotropic, linear elastic.

Nonlinearity of the system model may come in form of large rotations, material, geometric or contact. This is accounted for using iterative methods such as Netwon–Raphson or modified Newton.

For time-dependent numerical solutions, a time integration routine is needed. For an acceleration-based solving routine, the system acceleration is solved for at each time step and integrated to velocity and positions. The generalized-α method [5] is a generalization of several common integration methods and utilizes a predictor–corrector scheme [26].

The solving of the governing equations is referred to as primal analysis to distinguish it from sensitivity analysis. This three-module solving approach for the primal analysis can then be assembled for flexible multibody dynamics as the following (color scheme used throughout):

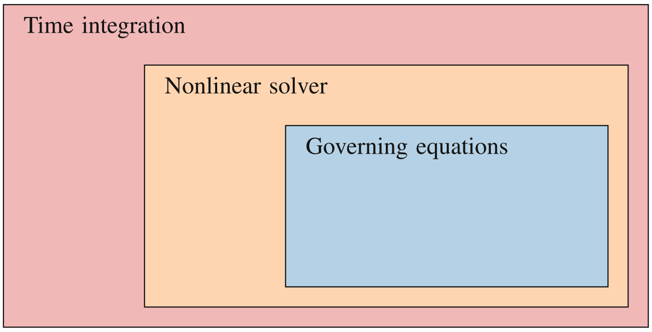


Fig. 3. Embedding of three-block solver scheme

Governing equations:

$$\text{force equilibrium:} \quad \underline{\underline{m}}\,\underline{\ddot{q}} + \underline{F}_{\text{int}} + \underline{\underline{\mathrm{J}\Phi}}\,\underline{\lambda} = \underline{F}_{\text{ext}} \tag{2}$$

$$\text{kinematic constraints:} \quad \underline{\Phi} = \underline{0} \tag{3}$$

$$\text{material, e.g. linear elasticity:} \quad \underline{\sigma}_i = \underline{\underline{E}}_i \underline{\varepsilon}_i, \ \forall i \text{ integrations points} \tag{4}$$

Nonlinear solver:

$$\text{e.g. Newton–Raphson:} \quad \underline{\underline{\ddot{\mathrm{J}}R}}\,\Delta\underline{\ddot{q}} + \underline{R} = \underline{0} \tag{5}$$

Time integration

$$\text{e.g. Generalized-}\alpha\text{:} \quad \underline{\ddot{q}} \rightarrow \underline{\dot{q}} \rightarrow \underline{q} \tag{6}$$

where $\underline{q}$ is position and its time derivatives velocities and accelerations denoted by an overdot, $\underline{\underline{m}}$ is the mass matrix, $\underline{F}_{\text{int}}$ is the internal force vector, $\underline{F}_{\text{ext}}$ is external force vector, $\underline{\Phi}$ is the vector of kinematic constraints, $\underline{\underline{\mathrm{J}\Phi}}$ is its Jacobian matrix (or positional derivatives) of the kinematic constraints.

This formulation allows the development of a single approach for kinematically constrained and unconstrained, linear and nonlinear models as well as dynamics, quasi-statics and statics of various mechanical analysis types (Table 1). This is essential for structural design optimization including the parametric nature of models and sensitivity analysis, this is especially the case in direct differentiation. This framework is implemented in the in-house mechanical solver SiMuLi (Simulation and sensitivity analysis for structural dynamics and Multibody dynamics in Lightweight engineering design, Italian: *you simulate*), which has been developed in [10–12, 14, 36].

Table 1. Application of three-block solver scheme to diverse mechanical simulations

	governing equation	nonlinear solver	time integration
multibody dynamics	🟦	🟦	🟦
nonlinear (implicit) dynamics	🟦	🟦	🟦
crash (explicit) dynamics	🟦		🟦
linear (implicit) dynamics	🟦		🟦
nonlinear (quasi-) statics	🟦	🟦	
eigenvalue vibration (modal analysis)	🟦		
eigenvalue buckling	🟦		
linear (quasi-) statics	🟦		

3 Design Sensitivity Analysis as Motor of Structural Design Optimization

Design sensitivity analysis is the centerpiece of gradient-based structural design optimization. The calculation results in the gradients of the system responses with respect to the vector design variables $\underline{x}$ and is denoted by the nabla symbol ∇,

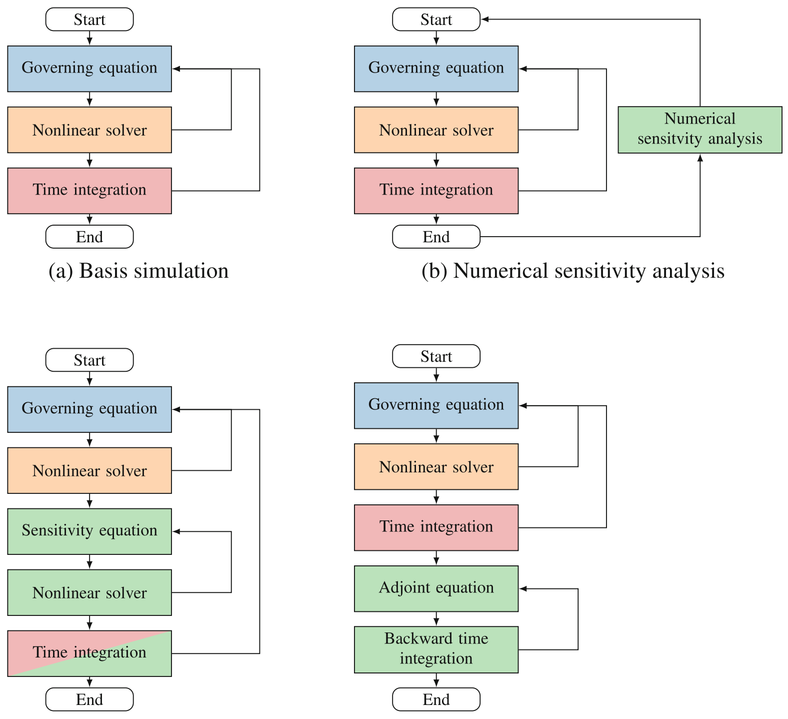
$$\underline{\nabla\left(\cdot\right)} = \frac{\mathrm{d}\left(\cdot\right)}{\mathrm{d}\underline{\mathsf{x}}}. \tag{7}$$

Several options exist for the calculation of design sensitivities, including numerical, analytical, complex step and automatic differentiation [18,23]. The architectures of the primary analysis, numerical sensitivity analysis, direct differentiation and the adjoint variable method are illustrated in Fig. 4b. For reasons of precision and computational effort, the emphasis is on analytical sensitivity analysis (i.e. direct differentiation and adjoint variable method), using numerical finite differencing for comparison. It should be noted that numerical sensitivity analysis, which is typically carried out via forward differencing, has the advantage of using the simulation as a black box, while direct differentiation and adjoint variable methods need access to the solver. The adjoint variable method needs a specially derived solver, see e.g. [3,4,7,17,24,25].

Adjoint variable method shows higher efficiency when the number of design variables is larger than the sensitivity responses needed, i.e. the number of optimization objectives and constraints, $n_{\mathsf{x}} > n_{\mathsf{f}} + n_{\mathsf{g}}$. Its use requires the implementation of an adjoint solver and, specifically in dynamics, backward time integration after the forward time integration of the primal analysis. On the other hand, direct differentiation utilizes the same equation structure and therefore solving routines for both primal and sensitivity analyses. The three-block solver scheme illustrates this procedure and breaks the solving routine into modules for the direct differentiation.

A semi-analytical approach is often preferred, which further reduces the implementation effort by numerical differentiating the components (partial derivatives) of the analytically total differentiated governing equations. The decoupling of the finite-element analysis from the multibody dynamic simulation is achieved via so-called multibody formulation specific invariants and their sensitivities [14,27,30]. Special attention must be paid to the sensitivity of time integration with flexible multibody dynamics, which is carried out in the example below using the generalized-α method [5,37] with a predictor–corrector scheme [26,35].

The direct differentiation of the three-block solving scheme of the primal analysis for flexible multibody dynamics (2)–(6) is the following:



(a) Basis simulation (b) Numerical sensitivity analysis

(c) Direct differentiation (d) Adjoint variable method

Fig. 4. Comparison of sensitivity analysis approaches with a basis generalized nonlinear dynamic simulation

Governing equations sensitivities:

$$\text{force equilibrium:}\quad \underline{\underline{m}}\,\underline{\underline{\nabla\ddot{q}}} + \underline{\underline{\nabla F}}_{\text{int}} + \underline{\underline{\mathrm{J}\Phi}}\,\underline{\underline{\nabla\lambda}} = \underline{\underline{F}}_{\text{pseudo}} \tag{8}$$

$$\text{where } \underline{\underline{F}}_{\text{pseudo}} = f\left(\underline{\underline{\nabla F}}_{\text{ext}}, \underline{\underline{\underline{\nabla m}}}, \underline{\underline{\underline{\nabla \mathrm{J}\Phi}}}\right) \tag{9}$$

$$\text{kinematic constraints:}\quad \underline{\underline{\nabla\Phi}} = \underline{0} \tag{10}$$

$$\text{material, e.g. linear elasticity:}\quad \underline{\underline{\nabla\sigma}}_i = \underline{\underline{\underline{\nabla E}}}_i\,\underline{\varepsilon}_i + \underline{\underline{E}}_i\,\underline{\underline{\nabla\varepsilon}}_i,\ \forall i \text{ integration points} \tag{11}$$

Nonlinear solver sensitivities:

$$\text{e.g. Newton–Raphson:}\quad \underline{\underline{\underline{\nabla\ddot{\mathrm{J}}R}}}\,\Delta\underline{\underline{\nabla\ddot{q}}} + \underline{\underline{\nabla R}} = \underline{0} \tag{12}$$

$$\rightarrow \underline{\underline{\ddot{\mathrm{J}}R}}\,\Delta\underline{\underline{\nabla\ddot{q}}} + \underline{\underline{\nabla R}} = \underline{0} \tag{13}$$

Time integration sensitivities:

$$\text{e.g. Generalized-}\alpha\text{:}\quad \underline{\underline{\nabla\ddot{q}}} \rightarrow \underline{\underline{\nabla\dot{q}}} \rightarrow \underline{\underline{\nabla q}} \tag{14}$$

It can be clearly seen that this mirrors that of the primary analysis and, therefore, the same routines can be used after extending these to solve for matrix-valued parameters, e.g. matrices $\underline{\underline{\nabla q}}$ of the sensitivity analysis in addition to vectors $\underline{q}$ of the primary analysis.

4 Morphing Wing Demonstrator

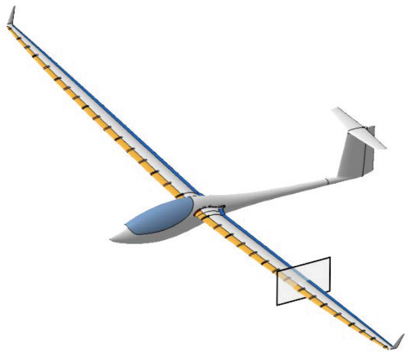
A morphing wing demonstrator for design optimization with flexible multibody simulation is developed in [9,10] based on the concept developed in [1,31–33] (Fig. 5). The previously developed model for sensitivity analysis is modified and extended here for the design formulation and used as basis for design optimization.

The forward section of the wing morphs from the undeformed, high-speed configuration to the deformed, low-speed configuration driven by a mechanism. A flexible multibody model is implemented with two-dimensional Euler–Bernoulli beam elements using the open-source software EasyBeam [8] in concert with rigid bodies for the driving mechanism. The rigid mechanism is loaded with a moment being applied to point A using a S-shaped load curve from 0 to 4000 N · mm [10]. The upper right node is fixed, while the lower right node is constrained vertically and in rotation. The rigid mechanism is connected with the flexible wing skin with revolute joints (hinges).

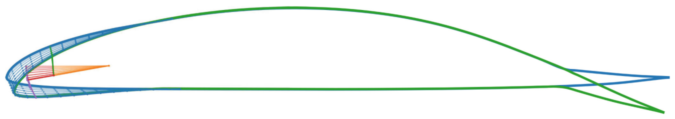
The lightweight engineering design formulation is applied in which the skin thickness of the wing h as well as the applied moment M_A and its position (x_A, y_A) are to be designed. The minimum mass design is to be found in which the wing is deformed to meet the high-speed configuration within a geometrical deviation limit and able to withstand repeated mechanical loading. Therefore, the limits are set for a global deviation measure, root mean square deviation at $\epsilon_{\text{rms}} = 1.5\,\text{mm}$, maximum local deviation of $\epsilon_{\text{local}} = 1.5\,\text{mm}$ and the limiting the stress at $\sigma_{\text{allow}} = 75\,\text{MPa}$. The maximum values of both local deviation and stress is ascertained by modified Kreisselmeier–Steinhauser functional aggregation [20,21] to avoid the noncontinuous max operator. The mathematical formulation of the optimization problem is

$$\begin{aligned}
&\min_{\underline{\mathsf{x}}\in\underline{\mathsf{X}}} \mathsf{f}\left(\underline{\mathsf{x}}\right), \qquad (15)\\
\text{where } &\mathsf{f}\left(\underline{\mathsf{x}}\right) = m\left(\underline{\mathsf{x}}\right),\\
\text{and } &\underline{\mathsf{x}} = \left[h\ M\ x_A\ y_A\right]^{\mathrm{T}},\\
\text{such that } &\mathsf{g}_{\sigma}\left(\underline{\mathsf{x}}\right) = \sigma_{\max}\left(\underline{\mathsf{x}}\right) - \sigma_{\text{allow}},\\
&\mathsf{g}_{\epsilon 1}\left(\underline{\mathsf{x}}\right) = \epsilon_{\text{rms}}\left(\underline{\mathsf{x}}\right) - \epsilon_{\text{rms,allow}},\\
&\mathsf{g}_{\epsilon 2}\left(\underline{\mathsf{x}}\right) = \epsilon_{\max}\left(\underline{\mathsf{x}}\right) - \epsilon_{\text{local,allow}},\\
&\underline{\mathsf{X}} = \left[\underline{\mathsf{x}}_L\ \underline{\mathsf{x}}_U\right],\\
\text{governed by DAE } &(2) - (4).
\end{aligned}$$

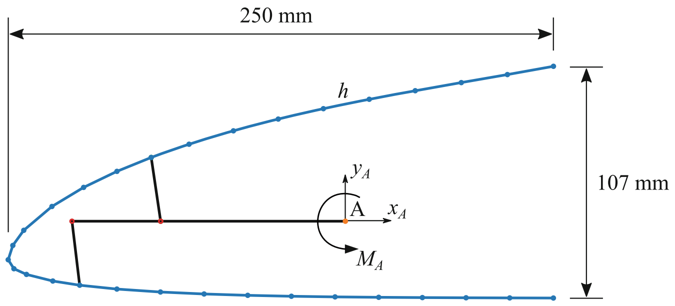
The second-order algorithm NLPQLP [6,28] is applied to this problem. As such the analytical gradients are provided and the design sensitivity analysis is carried



(a) Demonstration sailplane [1, 31, 32, 33], source: asg.ed.tum.de.



(b) Section of wing above showing morphing forward section with airfoil shapes for the undeformed low-speed configuration — and the deformed high-speed configuration — with movement of internal mechanism [9, 10].



(c) Flexible multibody model of morphing forward section (in undeformed state) with design variables shown for the design optimization problem. The model consisting of the flexible skin modeled with beam finite elements — and rigid driving mechanism —.

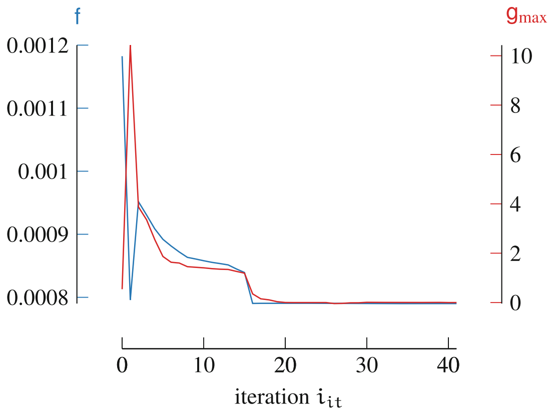
Fig. 5. Morphing wing demonstration example.

out using the direct differentiation of the three-block solving scheme introduced above. Benchmarked at the initial value as the computational time may vary within the design space, a time overview is given in Table 2. In addition to more precise sensitivities, analytical sensitivities overs a computation speedup with respect to numerical sensitivity analysis by forward finite differences.

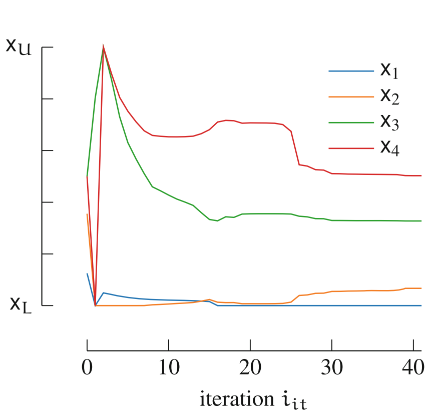
The problem converges in 42 iterations, requiring a total of 73 primary analyses and 42 sensitivity analyses in 3 h and 13 min on a quad-core laptop with an eleventh generation Intel Core i5 at 2.40 GHz. After starting in stress-violated

Table 2. Overview of computational effort for sensitivity analysis

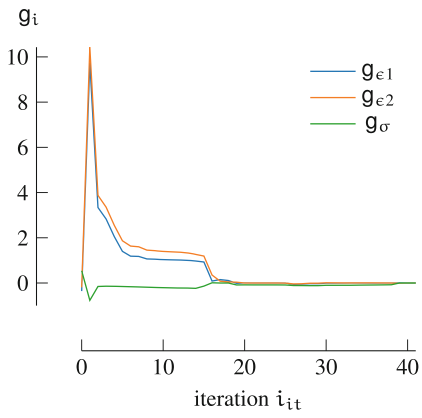
analysis	time [s]	speedup [×]
primary	126	–
primary and numerical sensitivity	630	–
primary with analytical sensitivity	160	3.9



(a) Objective function value and max constraint value



(b) Normalized design variable values



(c) Constraint function values

Fig. 6. Convergence plots for design optimization of morphing wing demonstration example.

infeasible space (approx. 80% violated), the algorithm reduces the skin thickness and applied moment resulting in deviation-violated design (approx. 10×). The algorithm then reduces the deviation violations to zero (see Fig. 6), resulting in the found optimum design in which all constraints are fulfilled.

Table 3. The design variables of the morphing wing demonstration example with initial values, upper and lower bounds as well as optimal values in MPa unit system.

design variable	symbol	x_0	x_L	x_U	x^*	unit
skin thickness	h	1.5	1.0	5.0	1.0	mm
applied moment	M_A	4200.0	1000.0	10 000.0	1601.95	N·mm
horizontal application point	x_A	0.0	-50.0	50.0	17.2432	mm
vertical application point	y_A	0.0	-25.0	25.0	-0.139269	mm

With the assumption that the nearly 4× speedup with analytical sensitivities holds throughout the design domain, one may postulate that this optimization with numerical sensitivities would take over 12 h. The reduced precision of the numerical sensitivities could further prolongate or impede the convergence. Analytical sensitivities can therefore be seen as an enabler of the design optimization of this example and further examples of this class of design problem.

5 Conclusion

The three-block solver scheme is introduced above as a general methodology for the sensitivity analysis of mechanical analysis via direct differentiation. Sensitivity analysis via direct differentiation exhibits the same equation pattern for all steps allowing the reuse of implemented routines for each block. Gradient-based design optimization with analytical sensitivities exploits this to reduce computational effort and, therefore, allow for larger problem sizes. On the contrary, numerical sensitivity analysis via forward finite differences requires $n_{\mathsf{x}} + 1$ evaluations and the choice of perturbation can prove challenging or even impossible to achieve proper precision. The morphing wing demonstration example is a relatively small problem, though it already shows a drastic reduction in computational effort, which will be further emphasized with growing problem size.

Acknowledgements. This developments of this work is supported by the project CRC 2017 TN2091 doloMULTI *Design of Lightweight Optimized structures and systems under MULTIdisciplinary considerations through integration of MULTIbody dynamics in a MULTIphysics framework* funded by the Free University of Bozen-Bolzano.

References

1. Achleitner, J., Rohde-Brandenburger, K., Rogalla von Bieberstein, P., Sturm, F., Hornung, M.: Aerodynamic design of a morphing wing sailplane. In: AIAA Aviation 2019 Forum. American Institute of Aeronautics and Astronautics, Reston, Virginia (2019). https://doi.org/10.2514/6.2019-2816
2. Bestle, D., Eberhard, P.: Analyzing and optimizing multibody systems. Mech. Struct. Mach. **20**(1), 67–92 (1992). https://doi.org/10.1080/08905459208905161

3. Boopathy, K., Kennedy, G.J.: Parallel finite element framework for rotorcraft multibody dynamics and discrete adjoint sensitivities. AIAA J. **57**, 1–14 (2019). https://doi.org/10.2514/1.j056585
4. Callejo, A., Sonneville, V., Bauchau, O.: Discrete adjoint method for the sensitivity analysis of flexible multibody systems. J. Comput. Nonlinear Dyn. (2018). https://doi.org/10.1115/1.4041237
5. Chung, J., Hulbert, G.M.: A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-α method. J. Appl. Mech. **60**(2), 371–375 (1993). https://doi.org/10.1115/1.2900803
6. Dai, Y.H., Schittkowski, K.: A sequential quadratic programming algorithm with non-monotone line search. Pac. J. Optim. **4**, 335–351 (2008)
7. Ebrahimi, M., Butscher, A., Cheong, H., Iorio, F.: Design optimization of dynamic flexible multibody systems using the discrete adjoint variable method. Comput. Struct. **213**, 82–99 (2019). https://doi.org/10.1016/j.compstruc.2018.12.007
8. Gufler, V., Wehrle, E.: EasyBeam: easy application for structural analysis with beams (2021). https://doi.org/10.5281/ZENODO.5674482
9. Gufler, V., Wehrle, E., Achleitner, J., Vidoni, R.: Flexible multibody dynamics and sensitivity analysis in the design of a morphing leading edge for high-performance sailplanes. In: ECCOMAS Multibody Dynamics Conference 2021. Budapest University of Technology and Economics (2021). https://doi.org/10.3311/eccomasmbd2021-203
10. Gufler, V., Wehrle, E., Achleitner, J., Vidoni, R.: A semi-analytical approach to sensitivity analysis with flexible multibody dynamics of a morphing forward wing section. Multibody Syst. Dyn. (2023). https://doi.org/10.1007/s11044-023-09886-9
11. Gufler, V., Wehrle, E., Vidoni, R.: Multiphysical design optimization of multibody systems: application to a Tyrolean weir cleaning mechanism. In: Niola, V., Gasparetto, A. (eds.) IFToMM ITALY 2020. MMS, vol. 91, pp. 459–467. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-55807-9_52
12. Gufler, V., Wehrle, E., Vidoni, R.: Analytical sensitivity analysis of flexible multibody dynamics with index-1 differential-algebraic equations and Baumgarte stabilization. Int. J. Mech. Control **24**(1), 3–14 (2023)
13. Gufler, V., Wehrle, E., Zwölfer, A.: A review of flexible multibody dynamics for gradient-based design optimization. Multibody Syst. Dyn. **53**(4), 379–409 (2021). https://doi.org/10.1007/s11044-021-09802-z
14. Gufler, V., Zwölfer, A., Wehrle, E.: Analytical derivatives of flexible multibody dynamics with the floating frame of reference formulation. Multibody Syst. Dyn. (2022). https://doi.org/10.1007/s11044-022-09858-5
15. Haug, E.J., Arora, J.S.: Applied Optimal Design: Mechanical and Structural Systems. Wiley, New York (1979)
16. Held, A.: On design sensitivities in the structural analysis and optimization of flexible multibody systems. Multibody Syst. Dyn. **54**, 53–74 (2022). https://doi.org/10.15480/882.3908
17. Held, A., Knüfer, S., Seifried, R.: Structural sensitivity analysis of flexible multibody systems modeled with the floating frame of reference approach using the adjoint variable method. Multibody Syst. Dyn. **40**, 287–302 (2016). https://doi.org/10.1007/s11044-016-9540-9
18. Hwang, J.T., Martins, J.R.R.A.: A computational architecture for coupling heterogeneous numerical models and computing coupled derivatives. ACM Trans. Math. Softw. **44**(4), 1–39 (2018). https://doi.org/10.1145/3182393

19. van Keulen, F., Haftka, R., Kim, N.: Review of options for structural design sensitivity analysis - part 1: linear systems. Comput. Methods Appl. Mech. Eng. **194**(30–33), 3213–3243 (2005). https://doi.org/10.1016/j.cma.2005.02.002
20. Kreisselmeier, G., Steinhauser, R.: Systematic control design by optimizing a vector performance vector. In: International Federation of Active Controls Symposium on Computer-Aided Design of Control Systems (1979). https://doi.org/10.1016/S1474-6670(17)65584-8
21. Lambe, A., Kennedy, G.J., Martins, J.: An evaluation of constraint aggregation strategies for wing box mass minimization. Struct. Multidisc. Optim. **55**, 257–277 (2016). https://doi.org/10.1007/s00158-016-1495-1
22. Margossian, C.C.: A review of automatic differentiation and its efficient implementation. WIREs Data Min. Knowl. Disc. **9**(4), e1305 (2019). https://doi.org/10.1002/widm.1305
23. Martins, J.R.R.A., Hwang, J.T.: Review and unification of methods for computing derivatives of multidisciplinary computational models. AIAA J. **51**(11), 2582–2599 (2013). https://doi.org/10.2514/1.j052184
24. Nachbagauer, K., Oberpeilsteiner, S., Sherif, K., Steiner, W.: The use of the adjoint method for solving typical optimization problems in multibody dynamics. J. Comput. Nonlinear Dyn. **10**(6), 061011–061011–10 (2015). https://doi.org/10.1115/1.4028417
25. Nejat, A.A., Moghadasi, A., Held, A.: Adjoint sensitivity analysis of flexible multibody systems in differential-algebraic form. Comput. Struct. **228**, 106–148 (2020). https://doi.org/10.1016/j.compstruc.2019.106148
26. Newmark, N.M.: A method of computation for structural dynamics. J. Eng. Mech. **85**(3), 67–94 (1959). https://doi.org/10.1061/jmcea3.0000098
27. Orzechowski, G., Matikainen, M.K., Mikkola, A.M.: Inertia forces and shape integrals in the floating frame of reference formulation. Nonlinear Dyn. **88**(3), 1953–1968 (2017). https://doi.org/10.1007/s11071-017-3355-y
28. Schittkowski, K.: NLPQLP: a Fortran implementation of a sequential quadratic programming algorithm with distributed and non-monotone line search. User's guide, version 4.0, Department of Computer Science, University of Bayreuth (2013)
29. Schmit, L.A.: Structural design by systematic synthesis. In: Proceedings of the Second Conference on Electronic Computation, vol. 960. ASCE (1960)
30. Shabana, A.A.: Dynamics of Multibody Systems, 3rd edn. Cambridge University Press, Cambridge (2013). https://doi.org/10.1017/cbo9781107337213
31. Sturm, F., Achleitner, J., Jocham, K., Hornung, M.: Studies of anisotropic wing shell concepts for a sailplane with a morphing forward wing section. In: AIAA Aviation 2019 Forum. American Institute of Aeronautics and Astronautics (2019). https://doi.org/10.2514/6.2019-2817
32. Sturm, F., Hornung, M.: Morphing shell design of a sailplane with a morphing forward wing section. In: XXXV OSTIV Congress (2021)
33. Sturm, F., Hornung, M.: Morphing forward wing section skin design for a sailplane considering wing bending. In: Deutsche Gesellschaft für Luft- und Raumfahrt (ed.) 33nd Congress of the International Council of the Aeronautical Sciences, Deutsche Gesellschaft für Luft- und Raumfahrt (2022)
34. Tortorelli, D.A., Michaleris, P.: Design sensitivity analysis: overview and review. Inverse Probl. Eng. **1**(1), 71–105 (1994). https://doi.org/10.1080/174159794088027573
35. Trier, S., Marthinsen, A., Sivertsen, O.: Design sensitivities by the adjoint variable method in nonlinear structural dynamics. In: SIMS Simulation Conference, Trondheim, Norway (1996)

36. Wehrle, E., Gufler, V.: Lightweight engineering design of nonlinear dynamic systems with gradient-based structural design optimization. In: Proceedings of the Munich Symposium on Lightweight Design 2020, pp. 44–57. Springer, Heidelberg (2021). https://doi.org/10.1007/978-3-662-63143-0_5
37. Wehrle, E., Gufler, V.: Analytical sensitivity analysis of dynamic problems with direct differentiation of generalized-α time integration (2023, submitted). https://doi.org/10.31224/osf.io/2mb6y

Influence of Weighted Gradients in Topology Optimization of Flexible Multibody Systems

Ali Azari Nejat, Alexander Held(✉), and Robert Seifried

Institute of Mechanics and Ocean Engineering, Hamburg University of Technology, Eißendorfer Straße 42, 21073 Hamburg, Germany
{ali.azari,alexander.held,robert.seifried}@tuhh.de

Abstract. This paper deals with a level set-based topology optimization of flexible bodies within flexible multibody systems. The required gradient is provided by an adjoint sensitivity analysis. In this work, a modified Solid Isotropic Material with Penalization (SIMP) method is applied, which helps to avoid numerical issues connected to spurious modes in dynamic problems. Nevertheless, the included parametrization factors lead to a weighted gradient. To test the validity of corresponding optimization results, differently weighted gradients are then used within the topology optimization of a flexible piston rod in a slider-crank mechanism. The obtained results show, among others, the range of appropriate weightings in the studied example.

1 Introduction

In the context of virtual prototyping, the method of multibody systems is often used in the modeling of dynamic systems undergoing large nonlinear working motions, such as in machine dynamics, robotics, or vehicle dynamics. In modern lightweight designs or fast-moving machines, the elastic deformation of the bodies may not be neglected anymore and requires the use of the method of flexible multibody systems, see [14,20]. As long as these elastic deformations remain small, the floating frame of reference method is, in many cases, the most efficient approach, see [11,13].

In the design process of lightweight machines and structures, topology optimization is becoming increasingly popular. However, especially in the design of members of flexible multibody systems, topology optimization is a challenging task. This results from the nonlinearity of the governing equations of motion and the dependency of the objective function on time, state, and design variables. Hereby most challenging in the optimization is the computation of the gradient of the parametrized flexible multibody system; see also [6,17]. In this work, a time-continuous adjoint sensitivity analysis is used, [4,7]. The basis of the topology optimization used in this work is a level set method taken from [3,21]. Hereby, the underlying finite element (FE) model of the flexible body is parametrized by a modified Solid Isotropic Material (SIMP) as suggested in [5]. Thus, numerical difficulties linked to spurious modes can be avoided.

The material parametrization factors lead to a weighted gradient. As the main contribution of this work, different factors are tested to calculate the weighted gradient within a level set-based topology optimization of a flexible piston rod in a slider-crank

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 30–39, 2024.
https://doi.org/10.1007/978-3-031-50000-8_3

mechanism. A comparison of the optimization results reveals the appropriate range of weighting factors for the studied mechanism.

The remainder of this work is organized as follows: In Sect. 2, the floating frame of reference method is briefly described. The considered objective function and time-continuous adjoint sensitivity analysis for an exact gradient calculation are addressed in Sect. 3. The application example is given in Sect. 4. Moreover, in Sect. 5, the chosen parametrization law is introduced. In Sect. 6, the corresponding optimization results are discussed. Lastly, Sect. 7 concludes with a summary of this work.

2 Floating Frame of Reference Formulation

The floating frame of reference formulation is used to model flexible bodies. In this approach, the reference frame of the flexible body is performing a large motion, while the small elastic deformation is described within this reference frame. Then the elastic deformation $\boldsymbol{u}_\mathrm{P}$ of a point P of the body is approximated by

$$\boldsymbol{u}_\mathrm{P} \approx \boldsymbol{\Phi}_\mathrm{P} \boldsymbol{q}_\mathrm{e}. \tag{1}$$

In Eq. (1), matrix $\boldsymbol{\Phi}_\mathrm{P}$ contains global shape functions and $\boldsymbol{q}_\mathrm{e}$ is the vector of elastic coordinates. The global shape functions are here obtained by modal reduction of the underlying FE-model of the flexible body, see [12]. It is noted that the underlying FE model of the flexible body is also used within the topology optimization process.

3 Objective Function and Gradient Calculation

In the current topology optimization, the global strain energy of the considered flexible body is minimized. To formulate a scalar ψ from the time-dependent global strain energy, which is often referred to as compliance, it is firstly integrated over the simulation period as

$$\psi = \int_{t_0}^{t_1} \boldsymbol{q}_\mathrm{e}^\top \boldsymbol{K}_\mathrm{e} \boldsymbol{q}_\mathrm{e} \, \mathrm{d}t. \tag{2}$$

from the initial time t_0 to the final time t_1. In Eq. (2), $\boldsymbol{K}_\mathrm{e}$ is the reduced stiffness matrix of the underlying FE-model of the flexible body.

For an efficient topology optimization, the gradient of the objective function ψ

$$\nabla \psi := \frac{\mathrm{d}\psi}{\mathrm{d}\boldsymbol{x}}, \tag{3}$$

containing the derivatives of the objective function ψ with respect to the design variables $\boldsymbol{x} \in \mathbb{R}^n$ is needed. In the considered SIMP approach, the design variables are density-like parameters, which correspond to the material filling of n finite elements.

The applied adjoint sensitivity analysis for the gradient calculation consists of two steps: first, a set of adjoint differential equations have to be derived from the system equations of the flexible multibody model and solved for so-called adjoint variables.

Then, the gradient can be computed from the adjoint variables and the derivatives of the system equations with respect to the design variables. For a detailed description of the adjoint method in rigid and flexible multibody dynamics, interested readers are referred to [4,7].

It should be noted that in the established level set-based topology optimization, which follows the procedure described in [3,21], the negative gradient

$$\widetilde{\nabla}\psi := -\nabla\psi \tag{4}$$

is generally employed.

4 Application Example

The planar flexible slider-crank mechanism shown in Fig. 1 is used as an application example. Similar testing models are applied in [7–9,16,18]. Here, the crank is assumed to be rigid and has length 100mm. The piston rod is flexible and has length 400mm, width 40mm, and thickness 10mm. Its material properties are given by density $\rho_0 = 8750\,\mathrm{kg/m^3}$, Young's modulus $E_0 = 50\,\mathrm{GPa}$ and Poisson's ratio $\nu = 0.3$.

In the underlying FE-model, the piston rod is discretized by 200×20 planar bilinear elements. The interface elements of the flexible rod are assumed to be rigid, and, thus, the design domain consists of 198×20 elements. The slider-crank mechanism moves in the horizontal plane.

For the optimization, a transient analysis within a time period of $T = 2\,\mathrm{s}$ is performed. In this time span, the crank angle is prescribed by a rheonomic constraint, whose definition can be found in [2] and which is shown in Fig. 2.

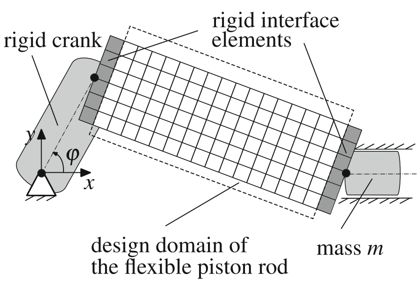


Fig. 1. Flexible slider-crank mechanism

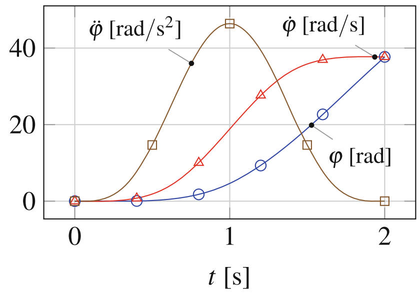


Fig. 2. Crank motion

5 Modified SIMP Method

In topology optimization, the design domain D is typically discretized by a finite element model. Thereby, the parametrization of the material properties of the finite elements is a crucial task, since it has a significant influence on the model order reduction, and thus, on the calculated gradient and corresponding optimization results. Different parametrization laws have been developed and used for structural and topology optimization of eigenvalue, vibration and transient problems, see for instance [1,5,8,10]. The applied parametrization law in the current work is described in the following.

In SIMP methods, the density ρ_i and Young's modulus E_i of a design element i are penalized by the corresponding design variable x_i and an artificial power law. The influence of different SIMP laws on the eigenmodes of a test structure is studied in [8]. The analyses unveil that the modified SIMP law by Du and Olhoff [5], defined as

$$\rho_i = \begin{cases} x_i \rho_0 & \text{for} \quad 0.1 \leq x_i \leq 1, \\ c x_i^q \rho_0 & \text{for} \quad x_{\min} \leq x_i < 0.1, \end{cases} \tag{5a}$$

$$E_i = {x_i}^p E_0, \tag{5b}$$

is suited to avoid the occurrence of spurious modes in the model order reduction, see [10] for more details about spurious modes in eigenfrequency problems.

In this formulation, ρ_0 is the density and E_0 the Young's modulus of the chosen reference material. To avoid ill-conditioned finite element system matrices, a lower limit $x_{\min} = 0.01$ is introduced for all design variables x_i. In this method, the density of elements with a design variable x_i between 0.1 and 1 is penalized linearly, whereas the density of poorly filled elements with a material amount between $x_{\min} = 0.01$ and 0.1 is penalized by a factor c and the power-law x_i^q. The power q is typically set to 6, and the factor c is selected such that the density interpolation around $x_i = 0.1$ is continuous, here $c = 10^5$ is chosen. Besides, the Young's modulus E_i is penalized by the power-law x_i^p, where the power p is typically set to 3, see also [5] for the set of chosen parameters. Here, the exponentiated penalization of the Young's modulus helps to develop thin connections and struts with a higher stability. Moreover, choosing $q = 6$ and $p = 3$, in the poorly filled regions with $x_i = x_{\min}$, the penalization of the density ρ_i compared to the Young's modulus E_i is clearly stronger. In this way, the spurious modes can be avoided.

Based on [8], the SIMP approach (5) is a successful parametrization method for dynamic problems, which is preferred in this work. Nevertheless, other parametrization laws, such as C^1 continuous modified SIMP law from [5] and the rational approximation of material properties (RAMP) proposed in [15] are also available, and can be utilized and tested within the optimization process as well.

In the utilized adjoint sensitivity analysis, the partial derivatives of the density ρ_i and Young's modulus E_i with respect to the design variables are included. Based on the selected SIMP parametrization (5), these partial derivatives can be formulated as

$$\frac{\partial \rho_i}{\partial x_i} = \begin{cases} \rho_0 & \text{for} \quad 0.1 \leq x_i \leq 1, \\ c q x_i^{(q-1)} \rho_0 & \text{for} \quad x_{\min} \leq x_i < 0.1, \end{cases} \tag{6a}$$

$$\frac{\partial E_i}{\partial x_i} = p {x_i}^{(p-1)} E_0. \tag{6b}$$

It is worth mentioning that using the SIMP approach (5) with the partial derivatives from Eq. (6), the obtained gradient includes some irregularities, which are due to the insignificantly small value of $\frac{\partial \rho_i}{\partial x_i}$ in the poorly filled areas with $x_i = x_{\min}$ and also C^0 continuity of the interpolation of ρ_i at $x_i = 0.1$. To circumvent these issues, it is decided to use the adjoint sensitivity analysis only for calculation of the gradient over elements with a design variable $x_i \geq 0.1$. For low density areas with $x_i < 0.1$, radial basis functions such as multiquadratic (MQ) splines can be used to approximate the gradient value, see [19] for a description and an application of radial basis functions in level set-based optimizations. Since in this strategy, the gradient of poorly filled elements with $x_{\min} \leq x_i < 0.1$ are not obtained by the adjoint sensitivity analysis, numerical artifacts are less likely to occur. Besides a higher efficiency can be expected than in the case, where the gradient of all elements should be obtained.

The SIMP law (5) and similar ones are originally developed to continuously modify the material properties within the design domain. Nevertheless, the exponents in the power laws change also the ratio of the partial derivatives $\frac{\partial \rho_i}{\partial x_i}$ and $\frac{\partial E_i}{\partial x_i}$. In other words, the exponents can also be seen as weighting factors, which, among others, set the influence of structural inertia and elasticity on the gradient obtained by the adjoint sensitivity analysis. However, other weighting factors can also be used to set the influences independently of the chosen exponents. For instance, using the SIMP approach (5) and introducing the parameters s_ρ and s_E, the weighted partial derivatives $\frac{\partial \rho_i}{\partial x_i}$ and $\frac{\partial E_i}{\partial x_i}$ for elements with $0.1 \leq x_i \leq 1$ can be written as

$$\frac{\partial \rho_i}{\partial x_i} = s_\rho \rho_0, \tag{7a}$$

$$\frac{\partial E_i}{\partial x_i} = s_\text{E} {x_i}^{(p-1)} E_0. \tag{7b}$$

Setting $s_\rho = 1$ and $s_\text{E} = 3$, the partial derivatives for elements with $0.1 \leq x_i \leq 1$ are weighted similar to Eq. (6). Nevertheless, to show the influence of these weighting factors, in the following section also further combinations are tested within the optimization process of the application example.

6 Optimization Results

It is obvious that unequal values for s_ρ and s_E change the influence of structural inertia and stiffness in the adjoint sensitivity analysis. To indicate the influence of the introduced parameters on the optimization results, different combinations $\{s_\rho = 1, s_\text{E} = 1\}$ (example A), $\{s_\rho = 1, s_\text{E} = 2\}$ (example B), $\{s_\rho = 1, s_\text{E} = 3\}$ (example C), $\{s_\rho = 1, s_\text{E} = 4\}$ (example D), as well as $\{s_\rho = 0, s_\text{E} = 1\}$ (example E) are used within the level set-based topology optimization of the flexible piston rod. Since in the studied example the inertial loads are dominant, see also [7, 9], gradients obtained with $s_\rho > s_\text{E}$ are not appropriate for the studied optimization problem and lead to the loss of structural cohesion in the optimization process. Hence, such combinations are not considered here.

In all optimization examples, a completely filled initial design is chosen. The volume limit v_max is set to 0.4, and the design is developed within a chosen number $n_\text{I} = 100$ of optimization iterations. It should be noticed that the completely filled initial design considered for different examples in this work is the most general case without any preconditions. Though, there are no limitations on the choice of the initial design with a volume fraction $v_0 < 1$. Moreover, the number n_I of optimization iterations, and the chosen volume limit v_max can be freely changed if desired.

The problem definitions and optimization results of the mentioned examples are summarized in Table 1. The corresponding final designs are shown in Fig. 3. These designs are zero-level contour plots of the corresponding final level set functions, which are created by the function `contourf` in MATLAB. Furthermore, the optimization histories of these five examples are shown in Fig. 4.

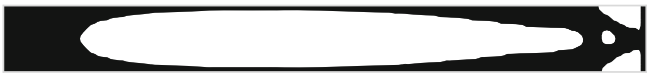
Table 1. Problem definitions and optimization results of demonstration examples in Sect. 6

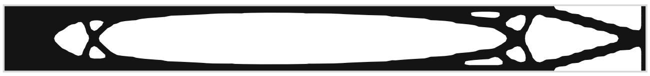
Example	A	B	C	D	E
Width/length ratio (FE-discretization)	1/10 (20×200)	1/10 (20×200)	1/10 (20×200)	1/10 (20×200)	1/10 (20×200)
Volume limit v_max	0.4	0.4	0.4	0.4	0.4
Successful iterations n_I	100	100	100	100	100
Final compliance ψ_end [Nmm]	0.062	0.048	0.048	0.051	0.055
Final volume fraction v_end	0.398	0.399	0.400	0.399	0.400
Selected s_ρ	1	1	1	1	0
Selected s_E	1	2	3	4	1

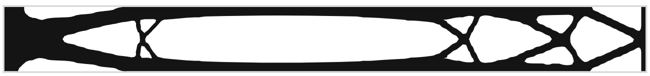
Among the presented results, the final design of example A, see Fig. 3a, has the lowest complexity and the highest final compliance, namely $\psi_\mathrm{end} = 0.062$ Nmm. This is due to the fact that the loading on the studied flexible piston rod is dominated by its inertia, see also [7,9]. Therefore, the equal parameters $\{s_\rho = 1, s_\mathrm{E} = 1\}$ lead to a high influence of partial derivatives $\frac{\partial \rho_i}{\partial x_i}$ on the gradient, and consequently imperfect optimization results.

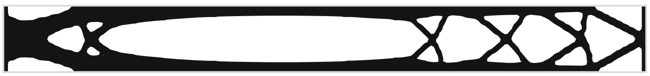
To decrease the influence of structural inertia on the gradient, the parameter s_E is increased to 2 in example B and to 3 in example C. In this way, meaningful optimization results are reached, see Fig. 3b and 3c. Compared to example A, the final designs of examples B and C are of higher complexity, which is due to the higher influence of structural elasticity on the negative gradient $\widetilde{\nabla}\psi$. The topologies of these two designs are though not similar. It can be argued that a change of s_ρ and s_E can also change the topology of the optimization results. The examples B and C provide the lowest final compliance among the five considered examples, namely $\psi_\mathrm{end} = 0.048$ Nmm. Hence, the corresponding combinations $\{s_\rho = 1, s_\mathrm{E} = 2\}$ and $\{s_\rho = 1, s_\mathrm{E} = 3\}$ are for the selected application example more suitable than the other mentioned cases.

Keeping $s_\rho = 1$ and increasing s_E stepwise from 2 to 4, it can be seen that in the corresponding final designs, the material amount is shifted from the left end to the middle area of the structure, which is connected to a slight increase of the final compliance from $\psi_\mathrm{end} = 0.048$ Nmm in examples B and C to $\psi_\mathrm{end} = 0.051$ Nmm in example D. Moreover, setting $s_\rho = 0$ and $s_\mathrm{E} = 1$, example E describes the case, where the influence of structural inertia in the negative gradient $\widetilde{\nabla}\psi$ is neglected, and only the influence


(a) ψ_{end}=0.062 Nmm, v_{end}=0.398 - example A


(b) ψ_{end}=0.048 Nmm, v_{end}=0.399 - example B


(c) ψ_{end}=0.048 Nmm, v_{end}=0.400 - example C


(d) ψ_{end}=0.051 Nmm, v_{end}=0.399 - example D

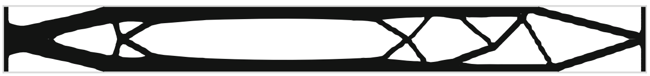
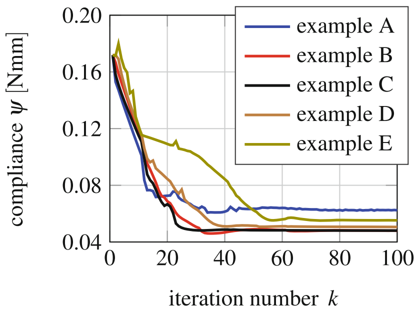
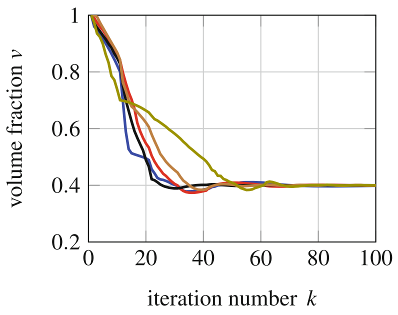

(e) ψ_{end}=0.055 Nmm, v_{end}=0.400 - example E

Fig. 3. Final designs of demonstration examples in Sect. 6



(a) Compliance



(b) Volume

Fig. 4. Optimization histories of demonstration examples in Sect. 6

of structural elasticity is considered. Compared to the examples B, C and D, the final design of example E has a higher compliance, $\psi_{\text{end}} = 0.055$ Nmm. This is mainly due to the shift of material amount from the left to the middle area rather than the lack of symmetry on the right end. The conclusion is that the influence of both the structural inertia and elasticity are in the adjoint sensitivity analysis significant. Though an appropriate weighting of them should be chosen, which depends, among others, on the studied dynamic problem.

For the five mentioned examples, the compliance ψ and volume fraction v over the iteration number k are shown in Fig. 4. Since the deformation of the flexible piston rod is mainly due to its own inertia, a decrease of the volume fraction v from 1 to 0.4 reduces in all examples the compliance ψ. Though the final compliances ψ_{end} are as mentioned before not similar.

The compliance histories in Fig. 4a include some peaks. These peaks arise, for instance, when the material removal from one iteration to the next is large, some struts are slightly interrupted or become too thin. Despite these minor irregularities in the compliance histories, the optimization process remains stable and converges for different examples to mentioned final compliances ψ_{end}. Thereby, only the compliance history in example A includes some small but frequent jags in the convergence area, which are due to the oscillation of material amount around the very thin longitudinal edges in the middle area. Such oscillations shall be handled, for instance, by amending the process parameters or choosing a finer FE-discretization.

Moreover, in all examples a convergence of the volume fraction v to the selected volume limit v_{max} can be observed. Nevertheless, the convergence speeds in different examples are not similar. This can, among others, be traced back to the different negative gradients $\widetilde{\nabla}\psi$ utilized in the optimization processes. If required, it is possible to change the convergence behavior by varying the process parameters.

7 Summary and Conclusion

In the current work, a level set-based topology optimization of flexible bodies in multibody systems modeled with floating frame of reference formulation is addressed. Thereby, a time-continuous adjoint sensitivity analysis is utilized for an exact gradient calculation, and the optimization is performed using a modified level set method. The presented approach is studied for the compliance minimization of a flexible piston rod within a slider-crank mechanism.

Using a modified SIMP parametrization method, the so-called spurious modes are avoided. Besides, the gradient in low density areas is approximated by radial basis functions to avoid numerical artifacts and inaccuracies in these regions and to reduce the computational effort of the adjoint sensitivity analysis. The SIMP parameters lead to weighted partial derivatives of the density and stiffness with respect to the design variables, which are included in the gradient calculation. To show the influence of weighted gradients on the optimization results, different weighting factors are tested within the gradient calculation. The corresponding optimization results for the considered application example reveal the fact that weighting the gradient is not a trivial task. Especially, in the studied dynamic example with a dominant inertial loading, the weighted gradients are useful to obtain more suited optimization results.

Acknowledgements. The authors would like to thank the German Research Foundation (DFG) for financial support of this work within a research project with the number 421344187.

References

1. Allaire, G., Jouve, F.: A level-set method for vibration and multiple loads structural optimization. Comput. Methods Appl. Mech. Eng. **194**(30–33), 3269–3290 (2005)
2. Azari Nejat, A., Held, A., Seifried, R.: A fully coupled level set-based topology optimization of flexible components in multibody systems. Struct. Multidiscip. Optim. **66**(7), 1–21 (2023)
3. Azari Nejat, A., Held, A., Trekel, N., Seifried, R.: A modified level set method for topology optimization of sparsely-filled and slender structures. Struct. Multidiscip. Optim. **65**(3), 1–22 (2022)
4. Bestle, D.: Analyse und Optimierung von Mehrkörpersystemen: Grundlagen und rechnergestützte Methoden. Springer (1994)
5. Du, J., Olhoff, N.: Topological design of freely vibrating continuum structures for maximum values of simple and multiple eigenfrequencies and frequency gaps. Struct. Multidiscip. Optim. **34**(2), 91–110 (2007)
6. Gufler, V., Wehrle, E., Zwölfer, A.: A review of flexible multibody dynamics for gradient-based design optimization. Multibody Sys.Dyn. **53**(4), 379–409 (2021)
7. Held, A., Knüfer, S., Seifried, R.: Structural sensitivity analysis of flexible multibody systems modeled with the floating frame of reference approach using the adjoint variable method. Multibody Sys.Dyn. **40**(3), 287–302 (2017)
8. Held, A., Nowakowski, C., Moghadasi, A., Seifried, R., Eberhard, P.: On the influence of model reduction techniques in topology optimization of flexible multibody systems using the floating frame of reference approach. Struct. Multidiscip. Optim. **53**(1), 67–80 (2016)
9. Moghadasi, A., Held, A., Seifried, R.: Topology optimization of members of flexible multibody systems under dominant inertia loading. Multibody Sys.Dyn. **42**(4), 431–446 (2018)
10. Pedersen, N.L.: Maximization of eigenvalues using topology optimization. Struct. Multidiscip. Optim. **20**(1), 2–11 (2000)
11. Schwertassek, R., Wallrapp, O.: Dynamik flexibler Mehrkörpersysteme: Methoden der Mechanik zum rechnergestützten Entwurf und zur Analyse mechatronischer Systeme. Grundlagen und Fortschritte der Ingenieurwissenschaften. Vieweg+Teubner Verlag (1999)
12. Schwertassek, R., Wallrapp, O., Shabana, A.A.: Flexible multibody simulation and choice of shape functions. Nonlinear Dyn. **20**(4), 361–380 (1999)
13. Seifried, R.: Dynamics of underactuated multibody systems: modeling, control and optimal design. Solid Mechanics and Its Applications. Springer Science & Business Media (2013)
14. Shabana, A.A.: Flexible multibody dynamics: review of past and recent developments. Multibody Sys.Dyn. **1**(2), 189–222 (1997)
15. Stolpe, M., Svanberg, K.: An alternative interpolation scheme for minimum compliance topology optimization. Struct. Multidiscip. Optim. **22**(2), 116–124 (2001)
16. Sun, J., Tian, Q., Hu, H.: Topology optimization based on level set for a flexible multibody system modeled via ANCF. Struct. Multidiscip. Optim. **55**(4), 1159–1177 (2017)
17. Tromme, E., Held, A., Duysinx, P., Brüls, O.: System-based approaches for structural optimization of flexible mechanisms. Arch. Comput. Methods Eng. **25**(3), 817–844 (2018)
18. Tromme, E., Tortorelli, D., Brüls, O., Duysinx, P.: Structural optimization of multibody system components described using level set techniques. Struct. Multidiscip. Optim. **52**(5), 959–971 (2015)
19. Wang, S.Y., Wang, M.Y.: Radial basis functions and level set method for structural topology optimization. Int. J. Numer. Meth. Eng. **65**(12), 2060–2090 (2006)

20. Wasfy, T.M., Noor, A.K.: Computational strategies for flexible multibody systems. Appl. Mech. Rev. **56**(6), 553–613 (2003)
21. Wei, P., Li, Z., Li, X., Wang, M.Y.: An 88-line MATLAB code for the parameterized level set method based topology optimization using radial basis functions. Struct. Multidiscip. Optim. **58**(2), 831–849 (2018)

Direct Differentiation of the Floating Frame of Reference Formulation via Invariants for Gradient-Based Design Optimization

Veit Gufler[1,3](✉), Erich Wehrle[1,2], and Andreas Zwölfer[3]

[1] Faculty of Engineering, Free University of Bozen-Bolzano, Universitätsplatz 1, 39100 Bozen, South Tyrol, Italy
Veit.Gufler@unibz.it

[2] Applied Research and Technology, Collins Aerospace, Munich, Germany
Erich.Wehrle@collins.com

[3] Department of Mechanical Engineering, Chair of Applied Mechanics, Technical University of Munich, TUM School of Engineering and Design, 85748 Garching bei München, Germany
Andreas.Zwoelfer@tum.de

Abstract. This paper applies the analytical differentiation of the equations of motion of flexible multibody systems modeled with the floating frame of reference formulation based on invariants. This leads to an efficient sensitivity analysis with the direct differentiation method and enables efficient design optimization of flexible multibody systems. The main results are the analytical derivatives of the equation terms of the floating frame of reference formulation in terms of inertia shape integrals or invariants. The introduced sensitivity analysis is applied and verified with a slider–crank mechanism modeled with beam elements. After numerical studies to assess the speedup, design optimization is carried out using the lightweight design formulation.

1 Introduction

Design optimization is a valuable tool for supporting engineers to achieve the best possible designs. Despite earlier publications in the field of design optimization of multibody systems [2,3,8,9,17] and more contemporaneously in [4,7,18,29], the optimization with flexible multibody dynamics remains a relevant topic of research.

The lightweight engineering design formulation is chosen to enable the *Virtual Circle of Lightweight Engineering Design* [27] in which mass is reduced, leading to saving in loads, drives as well as energy and cost, which in turn allows for further reduction in mass. To utilize lightweight engineering design with mechanical systems, gradient-based design optimization is used in concert with flexible multibody dynamics. Gradient-based optimization algorithms and analytical design sensitivities enable efficient optimization with flexible multibody dynamics and is reviewed in [15].

In this paper, analytical derivatives of the floating frame of reference formulation (FFRF) are used for both Jacobians as well as design sensitivities, which results in reduced computational effort as well as increased accuracy of the computed sensitivities. Faster computational times allow for larger models and higher number of design

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 40–51, 2024.
https://doi.org/10.1007/978-3-031-50000-8_4

variables. The increased accuracy typically leads to faster convergence of gradient-based optimization algorithms.

The methodology introduced below decouples the multibody simulation from the finite-element model for both primal analysis and sensitivity analysis. This is accomplished via use of the inertia shape integrals, referred to as invariants, and their sensitivities as introduced in [16]. It should be noted that this method will be shown with Euler–Bernoulli beams but the method is general in regard to finite-element type.

2 Flexible Multibody Dynamics Governing Equations with Invariants

The governing equations for flexible multibody dynamics is given by differential–algebraic equations and in this work expressed in index-1 form by

$$\underline{\underline{m}}\,\underline{\ddot{q}} + \underline{\underline{d}}\,\underline{\dot{q}} + \underline{\underline{k}}\,\underline{q} + \underline{\underline{J\Phi}}^{T}\underline{\lambda} = \underline{Q}_e + \underline{Q}_v, \tag{1}$$

$$\underline{\underline{J\Phi}}\,\underline{\ddot{q}} = \underline{Q}_c, \tag{2}$$

with the generalized positions $\underline{q}$, velocities $\underline{\dot{q}}$ and accelerations $\underline{\ddot{q}}$, the Lagrange multipliers $\underline{\lambda}$, the mass matrix $\underline{\underline{m}}$, the damping matrix $\underline{\underline{d}}$, the stiffness matrix $\underline{\underline{k}}$, the Jacobian matrix of constraints $\underline{\underline{J\Phi}}$, the generalized external forces $\underline{Q}_e$, the quadratic velocity forces $\underline{Q}_v$ and the right hand side of acceleration constraints $\underline{Q}_c$. The system matrices of $\underline{\underline{m}}$, $\underline{\underline{d}}$ and $\underline{\underline{k}}$ as well as the force vectors $\underline{Q}_e$, $\underline{Q}_v$ and $\underline{Q}_c$ are generally referred to as system parameters. Overdots represent the first $\dot{\bigcirc}$ and second $\ddot{\bigcirc}$ time derivatives, single underlined symbols $\underline{\bigcirc}$ represent vectors, double underlined symbols $\underline{\underline{\bigcirc}}$ represent matrices, triple underlined symbols $\underline{\underline{\underline{\bigcirc}}}$ represent three-dimensional expressions and overlined symbols $\overline{\bigcirc}$ represent local coordinates. The governing equations (1) and (2) are assembled in matrix form as

$$\begin{bmatrix} \underline{\underline{m}} & \underline{\underline{J\Phi}}^{T} \\ \underline{\underline{J\Phi}} & \underline{\underline{0}} \end{bmatrix} \begin{bmatrix} \underline{\ddot{q}} \\ \underline{\lambda} \end{bmatrix} = \begin{bmatrix} \underline{Q}_e + \underline{Q}_v - \underline{\underline{d}}\,\underline{\dot{q}} - \underline{\underline{k}}\,\underline{q} \\ \underline{Q}_c \end{bmatrix}. \tag{3}$$

The solving routine for flexible multibody dynamics and its design sensitivities used here is introduced in [10, 11, 26–28]. The time integration is carried out with the generalized-α method [5, 13, 28]. To avoid drift problems associated with index-1 differential–algebraic equations, Baumgarte stabilization is used [1, 13, 14]. The solving is referred to as the primal analysis to differentiate from the sensitivity analysis, introduced below.

In the present work, flexible multibody dynamics is modeled with FFRF [22, 23] and the generalized coordinates are given by

$$\underline{q} = \left[\, \underline{r}^{T} \ \underline{\beta}^{T} \ \underline{\overline{q}}_{f}^{T} \,\right]^{T}, \tag{4}$$

where $\underline{r}$ is the position vector from the inertial frame to the position of the floating frame, $\underline{\beta}$ is the orientation vector of the floating frame which dimension depend on the used orientation parametrization and $\underline{\overline{q}}_{f}$is the vector of flexible deformations, which

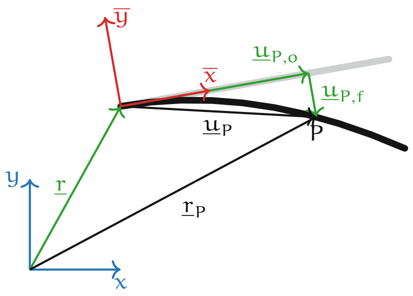


Fig. 1. Floating frame of reference coordinates

is comprised of flexible translations $\underline{u}_f$ and flexible rotations $\underline{\theta}_f$ for beam elements (Fig. 1). FFRF is characterized by a nonlinear mass matrix and a constant stiffness matrix.

The nonlinear FFRF mass matrix is given by

$$\underline{\underline{m}} = \begin{bmatrix} \underline{\underline{m}}_{tt} & \underline{\underline{m}}_{tr} & \underline{\underline{m}}_{tf} \\ & \underline{\underline{m}}_{rr} & \underline{\underline{m}}_{rf} \\ \text{sym.} & & \underline{\underline{m}}_{ff} \end{bmatrix}, \tag{5}$$

and is dependent on invariants (inertia shape integrals), which in turn depend on the linear-elastic structural finite-element model [23]. The individual terms of the FFRF mass matrix are [16, 19],

$$\underline{\underline{m}}_{tt} = \mathcal{I}_1 \underline{\underline{e}}, \tag{6}$$

$$\underline{\underline{m}}_{rr} = \underline{\underline{\overline{G}}}^T \left(\underline{\underline{\mathcal{I}}}_7 + \underline{\underline{I}}_8 \underline{\underline{\hat{\overline{q}}}}_f + \underline{\underline{\hat{\overline{q}}}}_f^T \underline{\underline{I}}_9 \underline{\underline{\hat{\overline{q}}}}_f \right) \underline{\underline{\overline{G}}}, \tag{7}$$

$$\underline{\underline{m}}_{ff} = \underline{\underline{\mathcal{I}}}_6, \tag{8}$$

$$\underline{\underline{m}}_{rf} = \underline{\underline{\overline{G}}}^T \left(\underline{\underline{\mathcal{I}}}_4 + \underline{\underline{\hat{\overline{q}}}}_f^T \underline{\underline{\mathcal{I}}}_5 \right), \tag{9}$$

$$\underline{\underline{m}}_{tf} = \underline{\underline{A}}\,\underline{\underline{\mathcal{I}}}_3, \tag{10}$$

$$\underline{\underline{m}}_{tr} = -\underline{\underline{A}} \left(\widetilde{\underline{\mathcal{I}}_2 + \underline{\underline{\mathcal{I}}}_3 \underline{\overline{q}}_f} \right) \underline{\underline{\overline{G}}}, \tag{11}$$

where $\underline{\underline{A}}$ is the rotation matrix, $\underline{\underline{G}}$ is the angular velocity matrix, $\mathcal{I}_i$ are the inertia shape integrals, $\underline{\overline{q}}_f$ are the flexible coordinates and $\underline{\underline{\hat{\overline{q}}}}_f$ is defined by

$$\underline{\underline{\hat{\overline{q}}}}_f = \begin{bmatrix} \underline{\overline{q}}_f & \underline{0} & \underline{0} \\ \underline{0} & \underline{\overline{q}}_f & \underline{0} \\ \underline{0} & \underline{0} & \underline{\overline{q}}_f \end{bmatrix}. \tag{12}$$

The inertia shape integrals are invariant in time and therefore are referred to as FFRF invariants, or simply invariants. For the full definition of the invariants, refer to [16, 19]. The sensitivity analysis is based on the invariant-based formulation and shown in the next section.

3 Invariant-Based Sensitivity Analysis

The direct differentiation of the governing equations (1) and (2) leads to

$$\begin{aligned}\underline{\underline{\underline{\nabla \mathrm{m}}}}\,\underline{\ddot{\mathrm{q}}}+\underline{\underline{\mathrm{m}}}\,\underline{\underline{\nabla \ddot{\mathrm{q}}}}+\underline{\underline{\underline{\nabla \mathrm{d}}}}\,\underline{\dot{\mathrm{q}}}+\underline{\underline{\mathrm{d}}}\,\underline{\underline{\nabla \dot{\mathrm{q}}}}+\underline{\underline{\underline{\nabla \mathrm{k}}}}\,\underline{\mathrm{q}}+\underline{\underline{\mathrm{k}}}\,\underline{\underline{\nabla \mathrm{q}}}+&\\ +\underline{\underline{\underline{\nabla \mathrm{J}\Phi^{\mathrm{T}}}}}\,\underline{\lambda}+\underline{\underline{\mathrm{J}\Phi^{\mathrm{T}}}}\,\underline{\underline{\nabla \lambda}}=\underline{\underline{\nabla \mathrm{Q}_e}}+\underline{\underline{\nabla \mathrm{Q}_\nu}},& \qquad (13)\\ \underline{\underline{\underline{\nabla \mathrm{J}\Phi}}}\,\underline{\ddot{\mathrm{q}}}+\underline{\underline{\mathrm{J}\Phi}}\,\underline{\underline{\nabla \ddot{\mathrm{q}}}}=\underline{\underline{\nabla \mathrm{Q}_c}},& \qquad (14)\end{aligned}$$

where the nabla operator ∇ represents the total derivative w.r.t. the design variables $\underline{\mathsf{x}}$

$$\underline{\nabla \bigcirc}=\frac{\mathrm{d}\bigcirc}{\mathrm{d}\underline{\mathsf{x}}}. \qquad (15)$$

The differentiated governing equations (13) and (14) are assembled into matrix form,

$$\begin{bmatrix}\underline{\underline{\mathrm{m}}} & \underline{\underline{\mathrm{J}\Phi^{\mathrm{T}}}}\\ \underline{\underline{\mathrm{J}\Phi}} & \underline{\underline{0}}\end{bmatrix}\begin{bmatrix}\underline{\underline{\nabla \ddot{\mathrm{q}}}}\\ \underline{\underline{\nabla \lambda}}\end{bmatrix}= \underbrace{\begin{bmatrix}\underline{\underline{\nabla \mathrm{Q}_e}}+\underline{\underline{\nabla \mathrm{Q}_\nu}}-\underline{\underline{\underline{\nabla \mathrm{m}}}}\,\underline{\ddot{\mathrm{q}}}-\underline{\underline{\underline{\nabla \mathrm{d}}}}\,\underline{\dot{\mathrm{q}}}-\underline{\underline{\mathrm{d}}}\,\underline{\underline{\nabla \dot{\mathrm{q}}}}-\underline{\underline{\underline{\nabla \mathrm{k}}}}\,\underline{\mathrm{q}}-\underline{\underline{\mathrm{k}}}\,\underline{\underline{\nabla \mathrm{q}}}-\underline{\underline{\underline{\nabla \mathrm{J}\Phi^{\mathrm{T}}}}}\,\underline{\lambda}\\ \underline{\underline{\nabla \mathrm{Q}_c}}-\underline{\underline{\underline{\nabla \mathrm{J}\Phi}}}\,\underline{\ddot{\mathrm{q}}}\end{bmatrix}}_{\text{pseudo load}}. \qquad (16)$$

The structure of the sensitivity analysis equations (16) mirrors that of the primal analysis (3) allowing for the use of the same solving routine.

The system equations are generally dependent on time, position, velocity and the design variables. The nonlinear FFRF mass matrix additionally has a dependency on the invariants $\mathcal{I}_\mathrm{i}$,

$$\underline{\underline{\mathrm{m}}}=\mathcal{F}\left(\mathcal{I}_\mathrm{i}\left(\underline{\mathsf{x}}\right),\underline{\mathrm{q}}\left(\underline{\mathsf{x}},\mathrm{t}\right)\right) \qquad (17)$$

and, therefore,

$$\underline{\underline{\underline{\nabla \mathrm{m}}}}=\mathcal{F}\Big(\mathcal{I}_\mathrm{i}\left(\underline{\mathsf{x}}\right),\underline{\mathrm{q}}\left(\underline{\mathsf{x}},\mathrm{t}\right),\nabla\mathcal{I}_\mathrm{i},\underline{\underline{\nabla \mathrm{q}}}\left(\mathrm{t}\right)\Big). \qquad (18)$$

The design sensitivities can be written via the chain rule for each time step as

$$\underline{\underline{\underline{\nabla \mathrm{m}}}}=\overbrace{\underline{\underline{\underline{\partial \mathrm{m}}}}}^{\frac{\partial \underline{\underline{\mathrm{m}}}}{\partial \mathcal{I}_\mathrm{i}}\frac{\mathrm{d}\mathcal{I}_\mathrm{i}}{\mathrm{d}\underline{\mathsf{x}}}}+\underline{\underline{\underline{\mathrm{J}\mathrm{m}}}}\,\underline{\underline{\nabla \mathrm{q}}}, \qquad (19)$$

where the operator ∂ represents the partial derivative w.r.t. the design variables,

$$\underline{\partial \bigcirc}=\frac{\partial \bigcirc}{\partial \underline{\mathsf{x}}}, \qquad (20)$$

and the operator J represents the partial derivative w.r.t. the generalized positions

$$\underline{\mathrm{J}\bigcirc}=\frac{\partial \bigcirc}{\partial \underline{\mathrm{q}}}. \qquad (21)$$

In previous studies [10–12,28], a semi-analytical approach is introduced in which the partial derivatives of the system parameters, e.g. $\underline{\underline{\partial m}}$, $\underline{\underline{\mathrm{J}m}}$ are approximated via finite differencing. This is extended by [11,16] in which an invariant-based approach is followed and applied here. The partial derivatives are directly differentiated and implemented to decrease computational effort and increase precision.

The invariant-based approach is applied analogously for all position- and velocity-dependent terms [16]. The partial derivative of the mass matrix is given by

$$\underline{\underline{\partial m}} = \begin{bmatrix} \underline{\underline{\partial m}}_{tt} & \underline{\underline{\partial m}}_{tr} & \underline{\underline{\partial m}}_{tf} \\ & \underline{\underline{\partial m}}_{rr} & \underline{\underline{\partial m}}_{rf} \\ \text{sym.} & & \underline{\underline{\partial m}}_{ff} \end{bmatrix}, \tag{22}$$

with its individual terms

$$\underline{\underline{\partial m}}_{tt} = \partial \mathcal{I}_1 \underline{\underline{e}}, \tag{23}$$

$$\underline{\underline{\partial m}}_{rr} = \underline{\overline{G}}^{T} \left(\underline{\underline{\partial \mathcal{I}}}_7 + \underline{\underline{\partial \mathcal{I}}}_8 \underline{\hat{q}}_f + \underline{\mathcal{I}}_8 \underline{\underline{\partial \hat{q}}}_f^T + \underline{\hat{q}}_f^T \underline{\underline{\partial \mathcal{I}}}_9 \underline{\hat{q}}_f + 2 \underline{\hat{q}}_f^T \underline{\mathcal{I}}_9 \underline{\underline{\partial \hat{q}}}_f^T \right) \underline{\overline{G}}, \tag{24}$$

$$\underline{\underline{\partial m}}_{ff} = \underline{\underline{\partial \mathcal{I}}}_6, \tag{25}$$

$$\underline{\underline{\partial m}}_{rf} = \underline{\overline{G}}^{T} \left(\underline{\underline{\partial \mathcal{I}}}_4 + \underline{\underline{\partial \hat{q}}}_f^T \underline{\mathcal{I}}_5 + \underline{\hat{q}}_f^T \underline{\underline{\partial \mathcal{I}}}_5 \right), \tag{26}$$

$$\underline{\underline{\partial m}}_{tf} = \underline{\underline{A}}\, \underline{\underline{\partial \mathcal{I}}}_3, \tag{27}$$

$$\underline{\underline{\partial m}}_{tr} = -\underline{\underline{A}} \left(\widetilde{\underline{\underline{\partial \mathcal{I}}}_2 + \underline{\underline{\partial \mathcal{I}}}_3 \underline{\overline{q}}_f + \underline{\mathcal{I}}_3 \underline{\underline{\partial \overline{q}}}_f} \right) \underline{\underline{G}}, \tag{28}$$

where the partial derivatives corresponding to the total derivatives of the invariants are calculated numerically for a semi-analytical approach. This procedure is followed for the differentiation of all system parameters, leading to a sensitivity method with high computational efficiency, while maintaining generality w.r.t. the design variables considered and outlined in detail in [16].

4 Numerical Results

The above described methodology is applied to a flexible slider–crank mechanism (Fig. 2) as introduced in [13,14]. The mechanism is modeled with Bernoulli–Euler beams, namely 10 elements for the crank shaft and 15 elements for the rod. A linear-elastic material model is assumed for a generic steel with a Young's modulus of

$$E = 210\,000\,\text{MPa}, \tag{29}$$

a Poisson's ratio of

$$\nu = 0.3, \tag{30}$$

a density of

$$\rho = 7.85 \times 10^{-9}\,\frac{\text{t}}{\text{mm}^3}, \tag{31}$$

and allowable stress including safety factors of

$$\sigma_{\text{allow}} = 100\,\text{MPa}, \tag{32}$$

where the consistent MPa system of units is used throughout.

The mechanism is fully extended in the initial position and the crank shaft rotates with high velocity of 5 000 rpm. The motion of the mechanism continues without external loads for a total duration of 0.025 s. The design variables are the height and width of each beam element (Fig. 4) while the system responses of interest are the stresses and system mass. First the speedups are shown with the invariant-based derivatives w.r.t. positions and velocities, i.e. Jacobians, and those w.r.t. design variables, i.e. design sensitivities. This is followed by a design optimization of the slider–crank mechanism using the invariant-based sensitivity analysis.

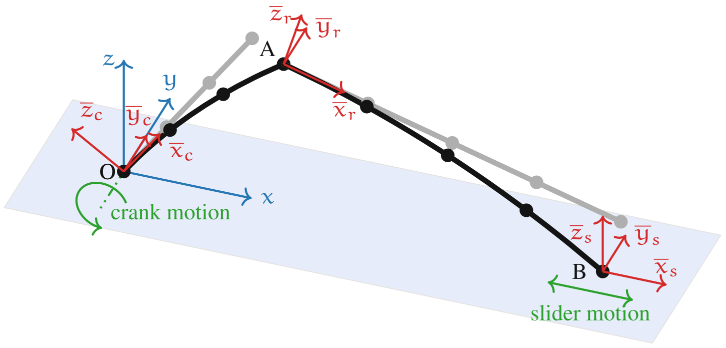


Fig. 2. Schematic of the slider–crank mechanism showing structure and FFRF coordinate system

4.1 Comparison of Numerical and Analytical Differentiation

In order to assess the computational effort advantages of the invariant-based derivatives, numerical and various levels of analytical derivatives are compared. The five different derivative methods are

- numerical total design derivatives, numerical Jacobians, i.e. ∇ numerical, $\mathcal{J}$ numerical;
- numerical total design derivatives, analytical Jacobians, i.e. ∇ numerical, $\mathcal{J}$ analytical;
- analytical total design derivatives, numerical design partial derivatives and Jacobians, i.e. ∇ analytical, ∂ numerical, $\mathcal{J}$ numerical;

- analytical total design derivatives, semi-analytical design partial derivatives via invariants and numerical Jacobians, i.e. ∇ analytical, ∂ numerical, $\mathcal{J}$ analytical;
- analytical design sensitivities, semi-analytical design partial derivatives via invariants and analytical Jacobians, i.e. ∇ analytical, ∂ semi-analytical, $\mathcal{J}$ analytical;

Numerical derivatives are calculated by forward finite differencing.

The analytical methods show significant reduction in computational effort as shown in Fig. 3. The full analytical method shows a nearly $50\times$ speedup with respect to the full numerical method. The analytical Jacobians are also of great importance to computational effort, though more so in the numerical sensitivity analysis in which it needs to be numerically computed for each of the $n_{\mathsf{x}}+1$ primal analyses.

4.2 Design Optimization

The introduced sensitivity analysis is verified and applied to the design optimization of a slider–crank mechanism with the lightweight design formulation. The lightweight engineering design formulation is chosen for which the mass is to be minimized while maintaining all requirements and constraints. The cross-sectional geometries of the crank and rod are to be designed, namely via the height h and width w, i.e. design variables $\underline{\mathsf{x}}$ (see Fig. 4) for a minimum mass m while remaining within upper and lower bounds ($\underline{\mathsf{x}}^{U}$ and $\underline{\mathsf{x}}^{L}$, respectively) and having a mechanical stress σ not exceeding the stress limit σ_{allow}. The standard formation of the optimal design problem is as follows:

$$\begin{aligned}
& \min_{\mathsf{x}\in\chi}\left\{\mathsf{f}(\underline{\mathsf{x}})\,|\,\underline{\mathsf{g}}(\underline{\mathsf{x}}) \overset{!}{\leqslant} 0\right\}, \\
\text{where}\quad & \mathsf{f}(\underline{\mathsf{x}}) = m_c + m_r, \\
\text{and}\quad & \underline{\mathsf{x}} = \left[h_{c,i}\ w_{c,i}\ h_{r,i}\ w_{r,i}\right]^{T}\ \forall i\,\text{elements}, \\
\text{such that}\quad & \underline{\mathsf{g}}(\underline{\mathsf{x}}) = \begin{cases} \frac{\sigma_{c,\max,i}}{\sigma_{\text{allow}}} - 1 \\ \frac{\sigma_{r,\max,i}}{\sigma_{\text{allow}}} - 1 \end{cases}, \\
\text{and}\quad & \chi = \left\{\mathsf{x}_i \in \mathbb{R}^{n_{\mathsf{x}}}\,|\,\mathsf{x}_i^{L} \leqslant \mathsf{x}_i \leqslant \mathsf{x}_i^{U}\right\}\ \forall i = 1,\ldots,n_{\mathsf{x}}, \\
\text{governed by}\quad & \text{DAE-1 (1–2).}
\end{aligned} \tag{33}$$

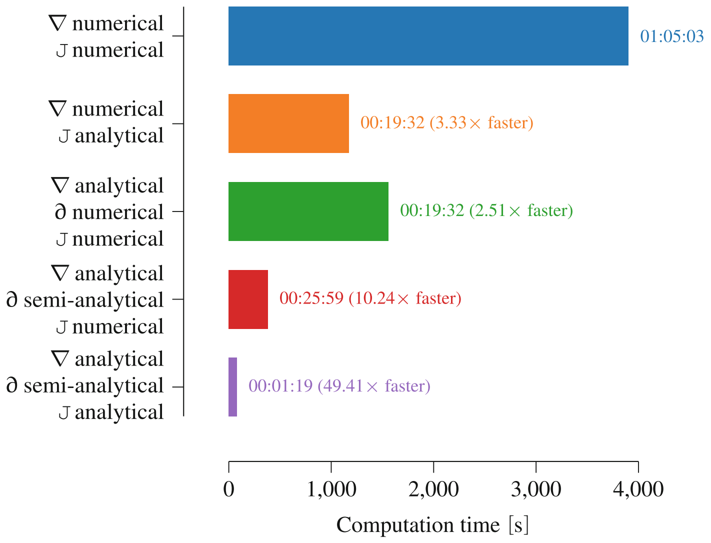


Fig. 3. Comparison of approaches for sensitivity analysis, where ∇ stands for the total design derivatives of the governing equation, ∂ stands for the design partial derivatives with respect to the design variables and J stands for the Jacobians, i.e. derivatives with respect to the position and velocity

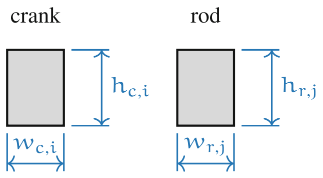
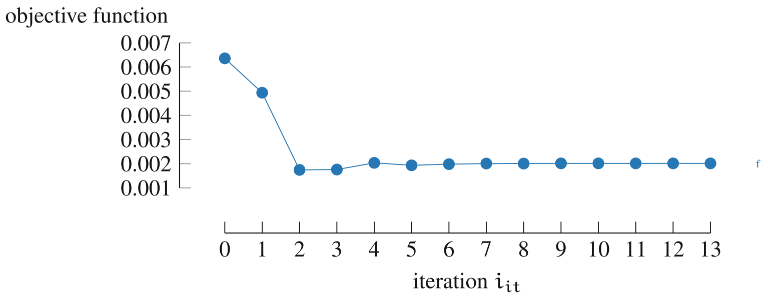


Fig. 4. Designable cross sections for the crank and rod of the slider–crank mechanism of Fig. 2

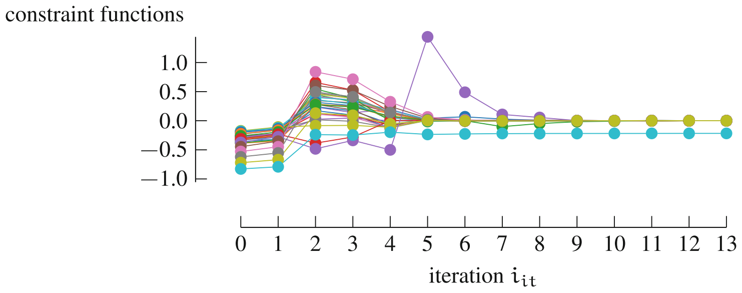
To solve this design problem the second-order algorithm NLPQLP [6, 21] in the DesOptPy framework [24, 25] via the pyOpt library [20] is used. The above described direct differentiation of FFRF is then applied.

The optimization converges rapidly starting from the upper bound to the optimal design (see Fig. 5),

$$\underline{x}_0 = \underline{x}^{\mathrm{U}}.$$

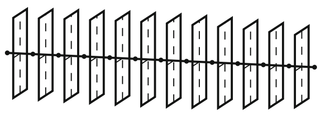


(a) Mass as objective function

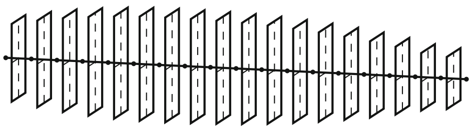


(b) Stress constraints

Fig. 5. Convergence of optimization for slider–crank with initial design at upper bound



(a) Crank shaft



(b) Rod

Fig. 6. Visualization of optimal design illustrated with slices of the cross-sectional geometries

The optimum is reached after approximately nine iterations and fulfilling the stop criteria after thirteen iterations to a final mass of approximately 0.002 t (2 kg). The appropriate area moment of inertia is found for the beam so that the most efficient use of material in which the widths are chosen to be at the lower bound and the heights are properly dosed. The numerical values of the optimal design are found in Table 1 and illustrated with slices of the cross-sectional geometry in Fig. 6.

Table 1. Design variable values for starting value, lower and upper bounds and optimal value (all values given in mm)

crank shaft design variables				
x	x_0	x^{L}	x^{U}	x^*
$w_{c,i}$	50	20	50	20.000
$h_{c,1}$	50	20	50	30.572
$h_{c,2}$	50	20	50	30.863
$h_{c,3}$	50	20	50	31.403
$h_{c,4}$	50	20	50	31.757
$h_{c,5}$	50	20	50	31.815
$h_{c,6}$	50	20	50	31.535
$h_{c,7}$	50	20	50	31.138
$h_{c,8}$	50	20	50	30.410
$h_{c,9}$	50	20	50	29.224
$h_{c,10}$	50	20	50	27.521

rod design variables				
x	x_0	x^{L}	x^{U}	x^*
$w_{r,i}$	50	20	50	20.000
$h_{r,1}$	50	20	50	30.384
$h_{r,2}$	50	20	50	34.251
$h_{r,3}$	50	20	50	37.161
$h_{r,4}$	50	20	50	38.910
$h_{r,5}$	50	20	50	39.763
$h_{r,6}$	50	20	50	39.925
$h_{r,7}$	50	20	50	39.492
$h_{r,8}$	50	20	50	39.017
$h_{r,9}$	50	20	50	37.955
$h_{r,10}$	50	20	50	36.313
$h_{r,11}$	50	20	50	34.105
$h_{r,12}$	50	20	50	31.315
$h_{r,13}$	50	20	50	27.872
$h_{r,14}$	50	20	50	23.595
$h_{r,15}$	50	20	50	20.000

5 Conclusion

In this work, an invariant-based formulation for analytical sensitivity analysis enables efficient design optimization. This framework is applied to a slider–crank mechanism with lightweight optimization formulation. The speedup in regards to the different sensitivity analysis types is compared showing a significant speedup using this invariant-based approach.

The outlook of this work includes further development and application towards large-scale design optimization. This includes increasing the number of elements and design variables, which includes shape and topology optimization. Although we have limited the scope here to include Euler–Bernoulli beams, the method is general in nature and as such further investigation is needed for other element types and formulations.

Acknowledgments. This work is supported by the project CRC 2017 TN2091 doloMULTI *Design of Lightweight Optimized structures and systems under MULTIdisciplinary considerations through integration of MULTIbody dynamics in a MULTIphysics framework* funded by the Free University of Bozen-Bolzano.

References

1. Baumgarte, J.: Stabilization of constraints and integrals of motion in dynamical systems. Comput. Methods Appl. Mech. Eng. **1**(1), 1–16 (1972). https://doi.org/10.1016/0045-7825(72)90018-7
2. Bestle, D.: Analyse und Optimierung von Mehrkörpersystemen. Springer, Berlin (1994). https://doi.org/10.1007/978-3-642-52352-6
3. Bestle, D., Eberhard, P.: Analyzing and optimizing multibody systems. Mech. Struct. Mach. **20**(1), 67–92 (1992). https://doi.org/10.1080/08905459208905161
4. Boopathy, K.: Adjoint based design optimization of systems with time dependent physics and probabilistically modeled uncertainties. Ph.D. thesis, Georgia Institute of Technology (2020). https://smartech.gatech.edu/handle/1853/63658
5. Chung, J., Hulbert, G.: A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-α method. J. Appl. Mech. **60**(2), 371–375 (1993). https://doi.org/10.1115/1.2900803
6. Dai, Y.H., Schittkowski, K.: A sequential quadratic programming algorithm with non-monotone line search. Pacific J. Optimiz. **4**(2), 335–351 (2008)
7. Ebrahimi, M., Butscher, A., Cheong, H., Iorio, F.: Design optimization of dynamic flexible multibody systems using the discrete adjoint variable method. Comput. Struct. **213**, 82–99 (2019). https://doi.org/10.1016/j.compstruc.2018.12.007
8. Etman, L., Van Campen, D., Schoofs, A.: Optimization of multibody systems using approximation concepts. In: IUTAM Symposium on Optimization of Mechanical Systems, Solid Mechanics and its Applications, vol. 43, pp. 81–88. Springer, Dordrecht (1996). https://doi.org/10.1007/978-94-009-0153-7_11
9. Etman, L.F.P., van Campen, D.H., Schoofs, A.J.G.: Design optimization of multibody systems by sequential approximation. Multibody Sys.Dyn. **2**(4), 393–415 (1998). https://doi.org/10.1023/A:1009780119839
10. Gufler, V., Wehrle, E., Achleitner, J., Vidoni, R.: Flexible multibody dynamics and sensitivity analysis in the design of a morphing leading edge for high-performance sailplanes. In: ECCOMAS Thematic Conference on Multibody Dynamics (2021). https://doi.org/10.3311/ECCOMASMBD2021-203. Budapest, Hungary (online)
11. Gufler, V., Wehrle, E., Achleitner, J., Vidoni, R.: A semi-analytical approach to sensitivity analysis with flexible multibody dynamics of a morphing forward wing section. Multibody Sys. Dyn. **58**(1), 1–20 (2023). https://doi.org/10.1007/s11044-023-09886-9
12. Gufler, V., Wehrle, E., Vidoni, R.: Multiphysical design optimization of multibody systems: application to a Tyrolean Weir cleaning mechanism. In: Niola, V., Gasparetto, A. (eds.) IFToMM ITALY 2020. MMS, vol. 91, pp. 459–467. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-55807-9_52
13. Gufler, V., Wehrle, E., Vidoni, R.: Sensitivity analysis of flexible multibody dynamics with generalized-α time integration and Baumgarte stabilization. In: Advances in Italian Mechanism Science, Mechanisms and Machine Science, vol. 122, pp. 147–155. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-10776-4_18
14. Gufler, V., Wehrle, E., Vidoni, R.: Analytical sensitivity analysis of flexible multibody dynamics with index-1 differential-algebraic equations and Baumgarte stabilization. Int. J. Mech. Control **24**(1), 3–14 (2023)
15. Gufler, V., Wehrle, E., Zwölfer, A.: A review of flexible multibody dynamics for gradient-based design optimization. Multibody Sys. Dyn. **53**(4), 379–409 (2021). https://doi.org/10.1007/s11044-021-09802-z
16. Gufler, V., Zwölfer, A., Wehrle, E.: Analytical derivatives of flexible multibody dynamics with the floating frame of reference formulation. Multibody Sys. Dyn. (2022). https://doi.org/10.1007/s11044-022-09858-5

17. Haug, E.J., Arora, J.S.: Design sensitivity analysis of elastic mechanical systems. Comput. Methods Appl. Mech. Eng. **15**(1), 35–62 (1978). https://doi.org/10.1016/0045-7825(78)90004-X
18. Held, A.: On design sensitivities in the structural analysis and optimization of flexible multibody systems. Multibody Sys. Dyn. **54**(1), 53–74 (2021). https://doi.org/10.1007/s11044-021-09800-1
19. Orzechowski, G., Matikainen, M.K., Mikkola, A.M.: Inertia forces and shape integrals in the floating frame of reference formulation. Nonlinear Dyn. **88**(3), 1953–1968 (2017). https://doi.org/10.1007/s11071-017-3355-y
20. Perez, R., Jansen, P., Martins, J.: pyOpt: a Python-based object-oriented framework for nonlinear constrained optimization. Struct. Multidiscip. Optim. **45**, 101–118 (2012). https://doi.org/10.1007/s00158-011-0666-3
21. Schittkowski, K.: NLPQLP: A Fortran Implementation of a Sequential Quadratic Programming Algorithm with Distributed and Non-Monotone Line Search, 4.2 edn. (2015). http://klaus-schittkowski.de/NLPQLP.pdf
22. Shabana, A.A.: Flexible multibody dynamics: review of past and recent developments. Multibody Sys. Dyn. **1**(2), 189–222 (1997). https://doi.org/10.1023/A:1009773505418
23. Shabana, A.A.: Dynamics of Multibody Systems, 5 edn. Cambridge University Press (2020). https://doi.org/10.1017/9781108757553
24. Wehrle, E.: Design optimization of lightweight space-frame structures considering crashworthiness and parameter uncertainty. Ph.D. thesis, Lehrstuhl für Leichtbau, Technische Universität München (2015). https://mediatum.ub.tum.de/doc/1244214/1244214.pdf
25. Wehrle, E.: DesOptPy: design optimization in python (2022). https://doi.org/10.5281/ZENODO.5908099. https://github.com/e-dub/DesOptPy
26. Wehrle, E.: Structural design optimization of dynamic systems: optimal lightweight engineering design via a three-block solver scheme for mechanical analysis. In: Optimal Design and Control of Multibody Systems, IUTAM 2022, IUTAM Bookseries. Springer (2022). https://doi.org/10.1007/978-3-031-50000-8_2
27. Wehrle, E., Gufler, V.: Lightweight engineering design of nonlinear dynamic systems with gradient-based structural design optimization. In: Proceedings of the Munich Symposium on Lightweight Design 2020, pp. 44–57. Springer, Berlin (2021). https://doi.org/10.1007/978-3-662-63143-0_5
28. Wehrle, E., Gufler, V.: Analytical sensitivity analysis of dynamic problems with direct differentiation of generalized-α time integration. (Submitted). https://doi.org/10.31224/osf.io/2mb6y
29. Zhang, M., Peng, H., Song, N.: Semi-analytical sensitivity analysis approach for fully coupled optimization of flexible multibody systems. Mech. Mach. Theory **159**, 104256 (2021). https://doi.org/10.1016/j.mechmachtheory.2021.104256

Comparison of Continuous and Discrete Adjoint Methods for Topology Optimization in Structural Dynamics

Timo Schmidt(✉), Justus Karnath, and Robert Seifried

Institute of Mechanics and Ocean Engineering, Hamburg University of Technology, Eißendorfer Straße 42, 21073 Hamburg, Germany
timo.schmidt@tuhh.de

Abstract. Semi-analytic adjoint methods can be seen as breakthrough for deterministic topology optimization in structural dynamics since such methods have reduced the computational costs. Nevertheless, the costs are still very high, so that further improvements are crucial. In this work, the concept of the continuous and the discrete adjoint method are applied in structural dynamics. Additionally, a modified discrete method, which is independent of the time discretization of the primal problem, is proposed. This approach results in further numerical savings, while the quality of the gradient does not suffer.

1 Introduction

The latest improvements in 3D-printing processes and today's need of lightweight designs have amplified the interest in topology optimization significantly. Topology optimization has been successfully applied in statics, however, the big breakthrough in dynamics is yet to come. This is mostly due to the computational costs that occur, if dynamics are considered. In contrast to statics, the sensitivity analysis of the objective results in a second differential equation that must be solved. These equations are derived with either the continuous (CAM) or discrete adjoint method (DAM). The DAM is based on the same discretization as the solution of the primal problem, so that the complete solution to the primal problem must be stored or recomputed. Applying this approach to real-world applications leads to high numerical costs and tremendous storage requirements, which make the optimization of dynamically loaded components very costly. In this work, the CAM and DAM are compared and an approximation of the discrete adjoint approach is proposed to reduce the storage requirements and the number of adjoint equations that must be solved.

All approaches are implemented in the topology optimization toolbox TOptiMuM, which uses the SIMP approach for parametrization [1], Sigmund's filter [6] and the Method of Moving Asymptotes [7] as optimizer.

2 Structural Dynamics

In structural dynamics, the balance of linear momentum for a single body is

$$\operatorname{div}(\boldsymbol{\sigma}) + \boldsymbol{b} = \tilde{\varphi}\dot{\boldsymbol{z}}, \tag{1}$$

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 52–61, 2024.
https://doi.org/10.1007/978-3-031-50000-8_5

where $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{b}$ are the volume forces, $\tilde{\varphi}$ is the local density and $\dot{\boldsymbol{z}}$ is the acceleration. Assuming linear elasticity and using the finite element method leads to a discrete equation of motion

$$\boldsymbol{ODE} := \boldsymbol{M}(\boldsymbol{\varphi})\dot{\boldsymbol{z}}(t) + \underbrace{\boldsymbol{K}(\boldsymbol{\varphi})\boldsymbol{y}(t) - \boldsymbol{f}_{\text{ext}}(\boldsymbol{y},t)}_{\boldsymbol{q}(\boldsymbol{y},t)} = \boldsymbol{0}, \tag{2}$$

where $\boldsymbol{M}(\boldsymbol{\varphi})$ is the mass matrix and $\boldsymbol{K}(\boldsymbol{\varphi})$ is the stiffness matrix of the linear elastic body. Both, the mass and stiffness matrix are obtained from a finite element analysis. For topology optimization, the elements density $\tilde{\varphi}_e$ and the elements Young's Modulus E_e are manipulated with a modified Solid Isotropic Material with Penalization (SIMP) approach [1,6]. They are

$$\tilde{\varphi}_e = \tilde{\varphi}_{\min} + \varphi_e(\tilde{\varphi}_{\max} - \tilde{\varphi}_{\min}) \quad \text{and} \quad E_e = E_{\min} + \varphi_e^p(E_{\max} - E_{\min}), \tag{3}$$

where the physical density $\tilde{\varphi}_e$ and Young's modulus E_e are functions of the normalized densities $\boldsymbol{\varphi} \in [\boldsymbol{0}, \boldsymbol{1}]$, which are the design variables of the later defined optimization problem. The penalization factor is usually chosen as $p = 3$. Further, the time-dependent nodal displacement field $\boldsymbol{y}(t)$, nodal velocity field $\boldsymbol{z}(t)$ and the external forces $\boldsymbol{f}_{\text{ext}}(\boldsymbol{y},t)$ are introduced. Equation (2) is solved for a given time interval that starts at t^0 and terminates at t^1.

2.1 Time Integration

The Newmark-β integration scheme [5] is used to solve Eq. (2). The scheme is

$$\boldsymbol{y}(t+\Delta t) = \boldsymbol{y}(t) + \Delta t \boldsymbol{z}(t) + \left(\frac{1}{2} - \beta\right)\Delta t^2 \dot{\boldsymbol{z}}(t) + \beta \Delta t^2 \dot{\boldsymbol{z}}(t+\Delta t), \tag{4}$$

$$\boldsymbol{z}(t+\Delta t) = \boldsymbol{z}(t) + (1-\gamma)\Delta t \dot{\boldsymbol{z}}(t) + \gamma \Delta t \dot{\boldsymbol{z}}(t+\Delta t), \tag{5}$$

with the timestep Δt. The method ranges from fully explicit schemes ($\beta = \gamma = 0$) to fully implicit schemes ($\beta = 0.5, \gamma = 1$). Here, the parameters are set to $\beta = 0.25$ and $\gamma = 0.5$. which leads to an unconditionally stable implicit method with second-order accuracy. In order to decrease the computational costs of the integration, the adaptive stepsize control of Zohdi [8] is implemented.

3 Topology Optimization

In topology optimization, a domain is discretized into finite elements and the densities of all elements $\boldsymbol{\varphi}$ are continuously varied between void ($\varphi_e = 0$) and full material ($\varphi_e = 1$). By updating the densities, the mechanical behavior of the total domain is changed, so that the performance of the domain or component can be determined using a cost function ψ. In this work, an integral cost function is considered. The optimization problem reads

$$\begin{cases} \min\limits_{\boldsymbol{\varphi}} \psi = \int\limits_{t^0}^{t^1} F(t, \boldsymbol{y}(t), \boldsymbol{z}(t), \dot{\boldsymbol{z}}(t), \boldsymbol{\varphi}) \mathrm{d}t \\ \text{s.t.} \begin{cases} \boldsymbol{ODE} = \boldsymbol{0} \quad \text{with} \quad \boldsymbol{q}(\boldsymbol{y}^0, 0) = \boldsymbol{q}^0 = \boldsymbol{0} \text{ and } \frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{q}(\boldsymbol{y}^0, 0)) = \dot{\boldsymbol{q}}^0 = \boldsymbol{z}^0 = \boldsymbol{0}, \\ V(\boldsymbol{\varphi}) - V_0 \leq 0, \boldsymbol{0} \leq \boldsymbol{\varphi} \leq \boldsymbol{1}. \end{cases} \end{cases} \tag{6}$$

In doing so, the time-dependent objective ψ is minimized and the considered motion of the domain is described by the $\boldsymbol{ODE}$ of Eq. (2), where the feasible initial state is defined by $\boldsymbol{q}(\boldsymbol{y}^0, 0) = \boldsymbol{0}$ and $\frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{q}(\boldsymbol{y}^0, 0))$. Additionally, the volume constraint V_0 is imposed and the design variables are constrained.

4 Sensitivity Analysis

A sensitivity analysis is necessary, because a deterministic topology optimization approach is used. A deterministic approach is chosen since solving the primal problem is very costly and can easily take up to several hours. In consequence, an adjoint method is used, to compute the gradient of the objective.

4.1 Continuous Adjoint Method

The main idea of the CAM is to start from the continuous optimization problem of Eq. (6). Next, using variational calculus, adjoint equations are derived such that all variations besides of the variation with respect to the design variables $\boldsymbol{\varphi}$ vanish. A detailed derivation can be found in [2].

The resulting gradient of the objective reads

$$\nabla \psi = -\left(\frac{\partial \boldsymbol{q}^0}{\partial \boldsymbol{\varphi}}\right)^{\mathrm{T}} \boldsymbol{\zeta}^0 - \left(\frac{\partial \dot{\boldsymbol{q}}^0}{\partial \boldsymbol{\varphi}}\right)^{\mathrm{T}} \boldsymbol{\eta}^0 + \int\limits_{t^0}^{t^1} \left[\frac{\partial F}{\partial \boldsymbol{\varphi}} - \left(\frac{\partial \boldsymbol{ODE}}{\partial \boldsymbol{\varphi}}\right)^{\mathrm{T}} (\boldsymbol{\nu} + \boldsymbol{\xi})\right] \mathrm{d}t, \tag{7}$$

where the adjoint vectors $\boldsymbol{\zeta}^0$, $\boldsymbol{\eta}^0$, $\boldsymbol{\nu}(t)$ and $\boldsymbol{\xi}(t)$ must be computed. The adjoints must satisfy

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} = \boldsymbol{\mu} - \dot{\boldsymbol{M}}\boldsymbol{\nu} + \left(\frac{\partial \boldsymbol{ODE}}{\partial \boldsymbol{z}}\right)^{\mathrm{T}} (\boldsymbol{\nu} + \boldsymbol{\xi}) - \frac{\partial F}{\partial \boldsymbol{z}}, \tag{8}$$

$$\dot{\boldsymbol{\mu}} = \left(\frac{\partial \boldsymbol{ODE}}{\partial \boldsymbol{y}}\right)^{\mathrm{T}} (\boldsymbol{\nu} + \boldsymbol{\xi}) - \frac{\partial F}{\partial \boldsymbol{y}}, \tag{9}$$

$$\boldsymbol{M}\boldsymbol{\xi} = \frac{\partial F}{\partial \dot{\boldsymbol{z}}}, \quad \left(\frac{\partial \dot{\boldsymbol{q}}^0}{\partial \boldsymbol{z}^0}\right)^{\mathrm{T}} \boldsymbol{\eta}^0 = \boldsymbol{M}^0 \boldsymbol{\nu}^0, \quad \left(\frac{\partial \boldsymbol{q}^0}{\partial \boldsymbol{y}^0}\right)^{\mathrm{T}} \boldsymbol{\zeta}^0 = \mu^0 - \left(\frac{\partial \dot{\boldsymbol{q}}^0}{\partial \boldsymbol{y}^0}\right)^{\mathrm{T}} \boldsymbol{\eta}^0, \tag{10}$$

where $\boldsymbol{\mu}(t)$ is an internal adjoint variable. Thus, Eq. (8) and Eq. (9) are first-order differential equations that must be solved backward in time. In doing so, the time-dependent adjoints at the termination time t^1 are given by

$$\boldsymbol{\mu}^1 = \mathbf{0} \quad \text{and} \quad \boldsymbol{M}^1 \boldsymbol{\nu}^1 = \mathbf{0}. \tag{11}$$

Here, Eq. (8) and Eq. (9) are solved with the implicit Heun integration scheme and the adaptive time-stepping is used. Hence, the primal and the continuous adjoint time-discretization are not the same and linear interpolation might be used to update state dependent parts during the backward integration.

In summary, the workflow of an optimization iteration consists of two independent steps. First, the primal problem defined in Eq. (2) is solved forward in time and the computed states $\boldsymbol{y}(t)$, $\boldsymbol{z}(t)$ and $\dot{\boldsymbol{z}}(t)$ are stored. Next, the adjoint Eqs. (8) and (9) are solved backward in time and the gradient of the objective with respect to the design variables is computed by Eq. (7).

4.2 Discrete Adjoint Method

In contrast to the CAM, the DAM couples solving the equation of motion and the sensitivity analysis. Therefore, the integral cost function is discretized into a fixed number of integration timesteps n forming the time-discretization for both, the primal problem and the discrete adjoint equations. The discrete objective reads

$$\psi \approx \psi_{\mathrm{D}} = \sum_{i=0}^{n-1} F(t_i, \boldsymbol{y}(t_i), \boldsymbol{z}(t_i), \dot{\boldsymbol{z}}(t_i), \boldsymbol{\varphi}) \Delta t_i, \quad \text{where} \quad \Delta t_i = (t_{i+1} - t_i), \tag{12}$$

and the Newmark-β integration scheme of the forward integration is included. Hence, instead of the equation of motion (2), the following set of equations must be satisfied

$$\boldsymbol{f}_1 = \boldsymbol{y}(t_i) - \boldsymbol{y}(t_{i-1}) - \Delta t_{i-1} \boldsymbol{z}(t_{i-1}) - \frac{\Delta t_{i-1}^2}{2} [(1-2\beta)\dot{\boldsymbol{z}}(t_{i-1}) + 2\beta \dot{\boldsymbol{z}}(t_i)] = \mathbf{0}, \tag{13}$$

$$\boldsymbol{f}_2 = \boldsymbol{z}(t_i) - \boldsymbol{z}(t_{i-1}) - \Delta t_{i-1}[(1-\gamma)\dot{\boldsymbol{z}}(t_{i-1}) + \gamma \dot{\boldsymbol{z}}(t_i)] = \mathbf{0}, \tag{14}$$

$$\boldsymbol{f}_3 = (\boldsymbol{M}\dot{\boldsymbol{z}})_{t_i} - \boldsymbol{q}(t_i, \boldsymbol{y}(t_i), \boldsymbol{z}(t_i)) = \mathbf{0}. \tag{15}$$

Thus, the discrete optimization problem reads

$$\begin{cases} \min\limits_{\boldsymbol{\varphi}} \psi_{\mathrm{D}} = \sum\limits_{i=0}^{n-1} F(t_i, \boldsymbol{y}(t_i), \boldsymbol{z}(t_i), \dot{\boldsymbol{z}}(t_i), \boldsymbol{\varphi}) \Delta t_i \\ \text{s.t.} \begin{cases} \boldsymbol{f}_i(\boldsymbol{x}_i, \boldsymbol{x}_{i-1}, \Delta t_{i-1}) = \mathbf{0} \text{ with } \tilde{\boldsymbol{q}}^0 = [\boldsymbol{q}^0, \dot{\boldsymbol{q}}^0, \ddot{\boldsymbol{q}}^0]^{\mathrm{T}} = \mathbf{0}, \\ V(\boldsymbol{\varphi}) - V_0 \leq 0, \ \mathbf{0} \leq \boldsymbol{\varphi} \leq \mathbf{1}, \end{cases} \end{cases} \tag{16}$$

where the state vector is $\boldsymbol{x}_i = [\boldsymbol{y}(t_i), \boldsymbol{z}(t_i), \dot{\boldsymbol{z}}(t_i)]^{\mathrm{T}}$ and the system equations are $\boldsymbol{f}_i = \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{x}_{i-1}, \Delta t_{i-1}) = [\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3]^{\mathrm{T}}$. A detailed derivation of the DAM is provided in [4].

Here, a brief sketch is given. The adjoint sensitivity analysis starts from extending the objective with the adjoint vectors $\tilde{\boldsymbol{\zeta}}_0$ and $\boldsymbol{p}_{i+1}$ such that both adjoint vectors can be arbitrary chosen. The extended objective reads

$$\psi_{\mathrm{D}} = \tilde{\boldsymbol{\zeta}}_0^{\mathrm{T}} \underbrace{\tilde{\boldsymbol{q}}^0}_{\mathbf{0}} + \sum_{i=0}^{n-1} \left[F(t_i, \boldsymbol{x}_i, \boldsymbol{\varphi}) \Delta t_i + \boldsymbol{p}_{i+1}^{\mathrm{T}} \underbrace{\boldsymbol{f}_{i+1}}_{\mathbf{0}} \right] . \tag{17}$$

The gradient of the objective with respect to the design variables follows as

$$\nabla \psi_{\mathrm{D}} = \left(\frac{\partial \tilde{\boldsymbol{q}}^0}{\partial \boldsymbol{\varphi}} \right)^{\mathrm{T}} \tilde{\boldsymbol{\zeta}}_0 + \sum_{i=0}^{n-1} \left[\Delta t_i \frac{\partial F_i}{\partial \boldsymbol{\varphi}} + \left(\frac{\partial \boldsymbol{f}_{i+1}}{\partial \boldsymbol{\varphi}} \right)^{\mathrm{T}} \boldsymbol{p}_{i+1} \right] , \tag{18}$$

where the adjoint vectors $\boldsymbol{p}_i$ are computed backwards. They must satisfy

$$\left(\frac{\partial \boldsymbol{f}_n}{\partial \boldsymbol{x}_n} \right)^{\mathrm{T}} \boldsymbol{p}_n = \mathbf{0} \quad \text{and} \quad \left(\frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{x}_i} \right)^{\mathrm{T}} \boldsymbol{p}_i = -\Delta t_i \left(\frac{\partial F_i}{\partial \boldsymbol{x}_i} \right) - \left(\frac{\partial \boldsymbol{f}_{i+1}}{\partial \boldsymbol{x}_i} \right)^{\mathrm{T}} \boldsymbol{p}_{i+1}, \tag{19}$$

and the discrete adjoint vector of the initial state $\tilde{\boldsymbol{\zeta}}_0$ must satisfy

$$\left(\frac{\partial \tilde{\boldsymbol{q}}^0}{\partial \boldsymbol{x}_0} \right)^{\mathrm{T}} \tilde{\boldsymbol{\zeta}}_0 = -\Delta t_0 \left(\frac{\partial F_0}{\partial \boldsymbol{x}_0} \right) - \left(\frac{\partial \boldsymbol{f}_1}{\partial \boldsymbol{x}_0} \right)^{\mathrm{T}} \boldsymbol{p}_1 . \tag{20}$$

The main difference to the CAM is that the used time discretization of the primal problem forms the basis of the sensitivity analysis. Consequentially, all computed states of the solution of the primal problem must be stored in order to compute the exact gradient. At the cost of extra computation, checkpoint methods [3] can be used to reduce storage requirements.

4.3 Modified Discrete Adjoint Method

The proposed modified discrete adjoint method (mDAM) is an approximation of the DAM. It is motivated by simulation results, where not all n states $\boldsymbol{x}_i$ of the solution of the primal problem had been stored or recomputed. These simulations have shown that both the objective ψ_{D} and the computed gradient $\nabla \psi_{\mathrm{D}}$ are not very sensitive to the time discretization, while the primal problem must be solved with the fine discretization of n states. Therefore, an equidistant adjoint time discretization of n_{red} states is created and only these states are stored and used to approximate the gradient of the objective $\nabla \psi$. The time discretization in the adjoint space is given by $t_k = t_0 + k \Delta t_{n_{\mathrm{red}}}$, where the number of stored states n_{red} is magnitudes smaller than the number of computed states n of the differential Eq. (2). Consequentially, the optimization problem is modified to

$$\begin{cases} \min\limits_{\boldsymbol{\varphi}} \psi_{\mathrm{D}} \approx \psi_{\mathrm{D,mod}} = \sum\limits_{k=0}^{n_{\mathrm{red}}-1} F(t_k, \boldsymbol{x}_k, \boldsymbol{\varphi}) \Delta t_{n_{\mathrm{red}}}, \quad \text{where} \quad n_{\mathrm{red}} << n \\ \text{s.t.} \begin{cases} \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{x}_{i-1}, \Delta t_{i-1}) = \mathbf{0} \; \forall i \in [0, n] \quad \text{with} \quad \boldsymbol{q}(\boldsymbol{y}^0, 0) = \mathbf{0} \text{ and } \boldsymbol{z}^0 = \mathbf{0}, \\ V(\boldsymbol{\varphi}) - V_0 \leq 0, \\ \mathbf{0} \leq \boldsymbol{\varphi} \leq \mathbf{1}. \end{cases} \end{cases} \tag{21}$$

In doing so, the state equation of the time discretization of the primal problem $\boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{x}_{i-1}, \Delta t_{i-1})$ are equal to zero, but the state equations of the adjoint time discretization $\boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}, \Delta t_{n_{\text{red}}})$ are not. The state equations $\boldsymbol{f}$ are based on two consecutive time instances t_{i-1} and t_i of the primal time discretization, which are not accessible anymore. However, since numerical solvers are used, the actual values of the state equations $\boldsymbol{f}$ will either way be close to zero but most likely not zero, so that the modified algorithm will work, if the state equations of the adjoint discretization are small enough

$$\boldsymbol{f}_k = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}, \Delta t_{n_{\text{red}}}) \leq \varepsilon, \quad \text{with} \quad \varepsilon << 1. \tag{22}$$

Hence, only the states of the solution of the primal problem $\boldsymbol{x}_i$, which just fulfill Eq. (22) must be stored. It turns out, that this approach does not result in a faster algorithm, because few entries of $\boldsymbol{f}_k$ become relatively large, even if just one timestep is neglected, so that almost all states will be stored again $n_{\text{red}} \approx n$.

Looking more closely into the values of $\boldsymbol{f}_k$ and the corresponding adjoint vectors $\boldsymbol{p}_k$ reveals that relatively large values of $\boldsymbol{f}_k$ correspond to relatively small values in the adjoint vector $\boldsymbol{p}_k$. This observation is considered by the following error estimator, which is based on the sum of the adjoint approach of the discrete objective of Eq. (17). Each addend of the sum is zero, if the time discretization of the primal is chosen, meaning that the standard DAM is applied. However, if the mDAM is used, the addends are not zero and must be sufficiently small for accurate results. The extended objective with the adjoint sum is

$$\psi \approx \psi_{\text{D,mod}} = \tilde{\boldsymbol{\zeta}}_0^{\text{T}} \underbrace{\tilde{\boldsymbol{q}}^0}_{\boldsymbol{0}} + \sum_{k=0}^{n_{\text{red}}-1} F(t_k, \boldsymbol{x}_k, \boldsymbol{\varphi}) \Delta t_{n_{\text{red}}} + \underbrace{\sum_{k=0}^{n_{\text{red}}-1} \boldsymbol{p}_{k+1}^{\text{T}} \underbrace{\boldsymbol{f}_{k+1}}_{\neq \boldsymbol{0}}}_{\text{adjoint sum}}. \tag{23}$$

In doing so, simulations have revealed that the reduced discretization is sufficient, if the adjoint sum is bounded by the error estimator

$$\frac{\sum_{k=0}^{n_{\text{red}}-1} \boldsymbol{f}_k^{\text{T}} \boldsymbol{p}_k}{\psi_{\text{D,mod}}} \leq \varepsilon, \quad \text{with} \quad \varepsilon << 1. \tag{24}$$

The drawback of this approach is, that the adjoint time discretization must be set before solving the differential equation and a good initial guess is needed. In structural dynamics, a good initial guess seems to be one or two orders of magnitude higher, than the time discretization of the primal problem. With the stored states $\boldsymbol{x}_k$, the adjoint Eq. (19) is computed to approximate the gradient of the adjoint. If the error estimator of Eq. (24) is satisfied, the gradient is excepted. Otherwise, the resolution of the adjoint time discretization must be increased and the state vectors of the new adjoint time discretization must be computed either by solving the primal problem again or by taking advantage of checkpoint approaches.

Summarizing, the modified discrete adjoint approach decouples the time discretization of the primal problem and the time discretization of the discrete adjoint computation. In doing so, only the states of the reduced time discretization $\boldsymbol{x}_k$ must be stored and

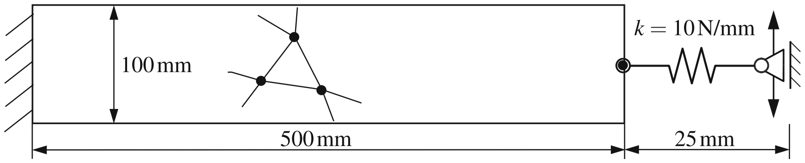


Fig. 1. Optimization domain of the cantilever beam.

are used to approximate the gradient of the objective. Thus, only n_{red} adjoint systems of equation must be solved instead of n for the standard discrete adjoint method. Consequentially, less storage is required and less systems of equations are solved, which leads to a better numerical performance.

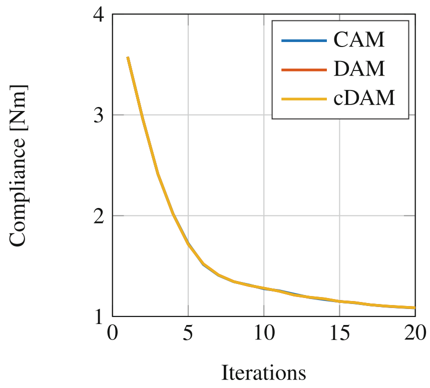
5 Numerical Example

The performance of all three adjoint methods is demonstrated using the simple cantilever beam example shown in Fig. 1a and the compliance is minimized

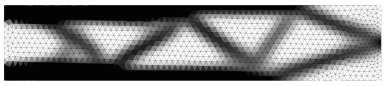
$$\psi_c = \int_{t^0}^{t^1} \boldsymbol{y}(t)^{\mathrm{T}} \boldsymbol{K}(\boldsymbol{\varphi}) \boldsymbol{y}(t) \mathrm{d}t \approx \psi_{D,c} = \sum_{i=0}^{n-1} \boldsymbol{y}_i^{\mathrm{T}} \boldsymbol{K}(\boldsymbol{\varphi}) \boldsymbol{y}_i \Delta t_i \approx \psi_{D,c,\text{red}}. \tag{25}$$

The Dirichlet boundary conditions are imposed on the left-hand side, while a spring is attached on the right-hand side. The spring is connected to a vertical oscillating sliding bearing, which oscillates with a frequency of 1 Hz and an amplitude of 50 mm. One period is considered, meaning that the starting time is $t^0 = 0$ s and the termination time is $t^1 = 1$ s. The absolute values of the spring force vary between approx. 250 N and 550 N. Note that, a non-linear problem must be solved to compute the initial state $\boldsymbol{x}^0$, since the spring force $\boldsymbol{f}_{\text{ext}}(\boldsymbol{y}^0, t^0)$ depends on the displacement $\boldsymbol{y}^0$.

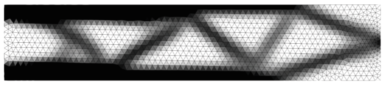
It is noticed, that the timestep control has no impact on the optimization, which is shown in Fig. 2a. Here, the optimizations using the CAM and DAM are compared to a reference optimization with a constant timestep of $10^{-}5$s (cDAM). The DAM is approx. 10 % faster than the CAM, even though both used the same time discretization for the primal problem. The reference optimization took ten times longer, than the CAM. Additionally, the optimized designs after 200 iterations using the CAM, DAM and mDAM are shown in Fig. 2 and they are identical.



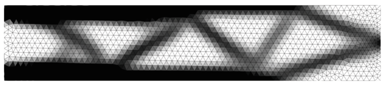
(a) Convergence of the compliance (coarse grid, 218 finite elements).



(b) Optimized design using CAM.



(c) Optimized design using DAM.



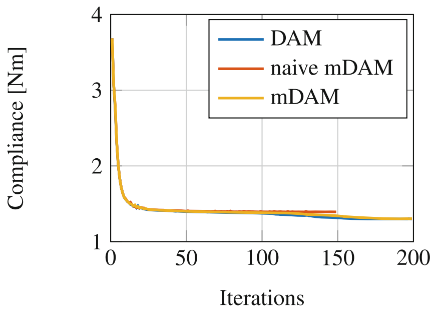
(d) Optimized design using mDAM.

Fig. 2. Optimizations computed with the continuous (CAM) and discrete (DAM) adjoint method using the timestep control and a reference discrete adjoint method (cDAM) with a constant time step.

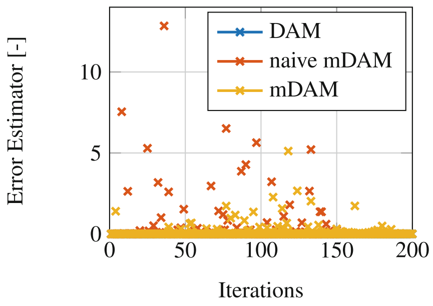
5.1 Optimization with the Modified Adjoint Method

In order to analyze the impact of the mDAM on the optimization process, the design domain is discretized into 2260 finite elements and three different discrete adjoint methods are used.

1. DAM - The previously validated discrete adjoint method.
2. naive mDAM - The modified DAM with a fixed adjoint time discretization.
3. mDAM - The proposed modified adjoint approach, where the error estimator of Eq. (24) is used with $\varepsilon = 0.01$. If the current adjoint time discretization is not sufficient, a finer adjoint time discretization is chosen and the differential equation is



(a) Convergence of the compliance.



(b) Error Estimator of Eq. (24).

Fig. 3. Comparison of the Discrete Adjoint Method (DAM), the naive modified DAM (naive mDAM) and the modified DAM (mDAM).

resolved. Otherwise, the computed derivative is accepted, the design update is computed and the time adjoint discretization is enlarged for the next iteration of the optimization.

The impact of the three methods is visualized in Fig. 3, where the convergence and the error estimator are shown. Looking at the convergence, the slope of the DAM and the mDAM are very similar, while the naive mDAM terminates at 150 iterations. This means that the maximum change of the design variables is less than 0.1 percent and a local minimum is assumed. At the same time, its objective value is larger than the objective values of the exact DAM or the mDAM. Thus, the naive approach can lead to convergence but the approximation of the gradient is inaccurate. The observed inaccuracy is also documented in the error estimator, since the error is significantly above $\varepsilon = 0.01$ at many iterations. However, the error of the mDAM is several time above the defined threshold as well, but if this happens, the adjoint time discretization is refined until the error criteria is met.

Coming to the storage requirements of the methods, the adjoint computation of the standard DAM is based on the time discretization of the primal problem. Therefore, all computed states of the forward integration must be stored and in consequence, for each state one adjoint equation must be solved. The actual number of states varies from iteration to iteration due to the adaptive time-stepping. Furthermore, the heterogeneous design vector leads to ill-conditioned mass and stiffness matrices, which has a significant impact on the needed number of states n. In this example, all optimizations start from a uniform density distribution and about $n = 50000$ states were computed during the forward integrations. Note that, the mDAM approaches solve the primal problem using the same forward time-discretization, while only fewer states are stored and less adjoint equations are solved. The naive mDAM is based on the adjoint discretization of $n_{\text{red}} = 1001$, whereas the average adjoint discretization of the mDAM is in the range of $n_{\text{red}} = [300, 5000]$ states. Hence, the mDAM approaches reduce the storage requirements tremendously.

Last, the average computational times per optimization iteration on a standard workstation have been 8236 s for the DAM, 1214 s for the naive mDAM and 2550 s for the mDAM. Hence, the optimization with the less accurate naive mDAM is 6.8x faster and the proposed mDAM is still 3.2x faster, while having a very similar convergence in comparison to the actual DAM.

6 Summary

In summary, the continuous and discrete adjoint method compute the gradient of an objective accurately. The backward integration of the continuous adjoint differential equations is numerically more expensive. Next, a modified discrete adjoint method has been proposed and compared to the standard discrete adjoint method. By applying this method, the storage requirements and the number of adjoint equations is reduced to a minimum. In doing so, the computation time was reduced by 69 %, while the error estimator guaranteed accurate gradient information during the optimization process.

Acknowledgments. The current work is a part of the research training group "Simulation-Based Design Optimization of Dynamic Systems Under Uncertainties" (SENSUS) funded by the state of Hamburg within the Landesforschungsförderung under project number LFF-GK11.

References

1. Bendsøe, M.P.: Optimal shape design as a material distribution problem. Struct. Optim. **1**(4), 193–202 (1989)
2. Bestle, D.: Analyse und Optimierung von MehrkÖrpersystemen: Grundlagen und rechnergestützte Methoden. Springer-Verlag (2013)
3. Giering, R., Kaminski, T.: Recipes for adjoint code construction. ACM Trans. Math. Softw. **24**(4), 437–474 (1998)
4. Lauß, T., Oberpeilsteiner, S., Steiner, W., Nachbagauer, K.: The discrete adjoint method for parameter identification in multibody system dynamics. Multibody Sys. Dyn. **42**(4), 397–410 (2018)
5. Newmark, N.M.: A method of computation for structural dynamics. J. Eng. Mech. ASCE **85**(EM3), 67–94 (1959)
6. Sigmund, O.: Morphology-based black and white filters for topology optimization. Struct. Multidiscip. Optim. **33**(4–5), 401–424 (2007)
7. Svanbergm, K.: The method of moving asymptotes - a new method for structural optimization. Inter. J. Numerical Methods Eng. **24**, 359–373 (1987)
8. Zohdi, T.I.: Charge-induced clustering in multifield particulate flows. Int. J. Numer. Meth. Eng. **62**(7), 870–898 (2005)

Derivation of Geometrically Parameterized Shell Elements in the Context of Shape Optimization

Manuel Vierneisel[1], Florian Geiger[2], Manfred Bischoff[2], and Peter Eberhard[1](✉)

[1] Institute of Engineering and Computational Mechanics, University of Stuttgart, Pfaffenwaldring 9, 70569 Stuttgart, Germany
{manuel.vierneisel,peter.eberhard}@itm.uni-stuttgart.de

[2] Institute for Structural Mechanics, University of Stuttgart, Pfaffenwaldring 7, 70569 Stuttgart, Germany
{geiger,bischoff}@ibb.uni-stuttgart.de

Abstract. The goal of this contribution is to extend an existing workflow for shape optimization with geometrically parameterized finite elements by implementing a parameterized shell element. The challenge is to find an element technology that is free of locking and hourglass-modes on the one hand, but its implementation must be simple enough so that it can contain the geometric parameters on the other hand. The derived element is used in a numerical optimization example. Therefore, a ribbed plate is used to show the whole workflow including the preparation of the system matrices depending on global design parameters, parametric model order reduction and shape optimization. A final validation is done using a commercial finite element software to compare the results.

1 Introduction

During the development cycle in mechanical or civil engineering the goal is to find a product that fulfills multiple requirements on deformations, stresses, or design. Since it is not clear from the beginning, which geometry fulfills all these specifications, several iterations with different shapes have to be made. The finite element method is a state of the art tool for modeling and analysis of such elastic structures. Typically, each modification requires some pre-processing, e.g., creation of the finite element mesh and, at least for complex systems, model order reduction. To avoid these time-consuming steps, a method combining geometrically parameterized finite elements and parametric model order reduction was developed in [11]. Since the existing method includes only hexahedron elements, the goal of this contribution is the extension of this approach to a geometrically parameterized four-node linear shell element.

The paper is structured as follows. Firstly, the necessary theoretical basics in the finite element method, model order reduction, and parametric model order reduction are revised. Secondly, different element technologies for a geometrically parameterized finite shell element are compared. Finally, the derived element is used in a shape optimization example and validated with results from a commercial finite element software.

2 Theoretical Background

This section summarizes the necessary information on the finite element method, model order reduction and its extension to parametric model order reduction.

2.1 Finite Element Method

The finite element method uses spatial discretization of structures to overcome the issue that for many problems in structural mechanics it is not possible to find an analytical solution. Details on the derivation of the finite element method can, e.g., be seen in [4] or [19]. This section focuses on a four node plane shell element with bilinear shape functions and three translational and three rotational degrees of freedom at each node. The formulation is based on the Reissner-Mindlin shell theory. The element mass matrix of the e-th element is

$$\boldsymbol{m}^e = \int_{-1}^{1}\int_{-1}^{1} \boldsymbol{N}^\mathsf{T} \begin{pmatrix} \rho t \boldsymbol{I}^{[3\times 3]} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \frac{\rho t^3}{12}\boldsymbol{I}^{[2\times 2]} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & f\frac{\rho t^3}{12} \end{pmatrix} \boldsymbol{N} \det(\boldsymbol{J}) \mathrm{d}\xi \mathrm{d}\eta \tag{1}$$

with the density ρ, the thickness of the element t, the identity matrices $\boldsymbol{I}$ and a drill factor f. The matrix $\boldsymbol{N} \in \mathbb{R}^{24\times 6}$ contains the shape functions that map the 24 node displacements to the rotational and translational element displacements, and ξ, η are the local curvilinear element coordinates. The Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{2\times 2}$ maps these local element coordinates to the Cartesian coordinates x and y. The element stiffness matrix for a pure displacement based element formulation can be calculated as

$$\boldsymbol{k}^e = \int_{-1}^{1}\int_{-1}^{1} \widetilde{\boldsymbol{B}}^\mathsf{T} \boldsymbol{E} \widetilde{\boldsymbol{B}} \det(\boldsymbol{J}) \mathrm{d}\xi \mathrm{d}\eta = \begin{pmatrix} \boldsymbol{k}_\mathrm{p}^e & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{k}_\mathrm{b}^e + \boldsymbol{k}_\mathrm{s}^e & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{k}_\mathrm{drill}^e \end{pmatrix}, \tag{2}$$

where $\widetilde{\boldsymbol{B}} \in \mathbb{R}^{24\times 8}$ is the strain-displacement matrix and $\boldsymbol{E} \in \mathbb{R}^{8\times 8}$ denotes the material matrix for linear isotropic plane stress. It can be seen that the in plane part of the stiffness matrix $\boldsymbol{k}_\mathrm{p}^e \in \mathbb{R}^{8\times 8}$ is decoupled from the bending and shear part of the stiffness matrix $\boldsymbol{k}_\mathrm{b}^e \in \mathbb{R}^{12\times 12}$ and $\boldsymbol{k}_\mathrm{s}^e \in \mathbb{R}^{12\times 12}$, which holds for plane elements. The stiffness matrix often includes an artificial drill stiffness $\boldsymbol{k}_\mathrm{drill}^e \in \mathbb{R}^{4\times 4}$ to avoid singular matrices. In this contribution, the drill stiffness is implemented as in [13]. The numerical integration of Eqs. (1) and (2) is typically done with Gauss-Legendre quadrature. For four node shell elements, it is common to either use so-called full numerical integration with two sampling points per direction or reduced integration with only one sampling point per direction as described in [4]. However, both integration methods have some drawbacks for a pure displacement based element formulation. Different forms of locking resulting in a too stiff element behavior occur in elements with full integration whereas reduced integration possibly leads to zero energy modes, also called hourglass modes, which produces singular stiffness matrices. There exist numerous methods trying to avoid locking and zero energy modes. An overview on this topic is given in [8]. In this contribution for the element implementation with full integration, the Enhanced Assumed Strain (EAS) Method developed in [18] with four extra parameters is used for the membrane part $\boldsymbol{k}_\mathrm{p}^e$ to avoid membrane locking and the Assumed Natural Strain (ANS) Method introduced in [5] is used to avoid transverse shear locking in $\boldsymbol{k}_\mathrm{s}^e$. According to [8], this combination is very popular and implemented in many finite element software tools. This element formulation is compared to an implementation with reduced integration using the hourglass stabilization method proposed in [14].

The assembly of the global mass $\boldsymbol{M} \in \mathbb{R}^{N\times N}$ and stiffness matrix $\boldsymbol{K} \in \mathbb{R}^{N\times N}$ of the whole structure results in the equation of motion

$$\begin{aligned} \boldsymbol{M}\ddot{\boldsymbol{q}}(t)+\boldsymbol{D}\dot{\boldsymbol{q}}(t)+\boldsymbol{K}\boldsymbol{q}(t) &= \boldsymbol{B}\boldsymbol{u}(t), \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{q}(t). \end{aligned} \tag{3}$$

Here, $\boldsymbol{q(t)} \in \mathbb{R}^N$ describes the displacements of all nodes. In system dynamics the external forces are often described as a product of the input matrix $\boldsymbol{B} \in \mathbb{R}^{N\times l}$ and the system inputs $\boldsymbol{u} \in \mathbb{R}^l$. Furthermore, a damping matrix $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ is often added to model the dissipation of energy. Later in this publication Rayleigh damping $\boldsymbol{D} = \alpha\boldsymbol{M} + \beta\boldsymbol{K}$ with damping parameters $\alpha, \beta \in \mathbb{R}$, see [20], is used, but the used methods are not limited to this kind of damping. The system outputs $\boldsymbol{y}(t) \in \mathbb{R}^k$ are obtained with the output matrix $\boldsymbol{C} \in \mathbb{R}^{k\times N}$ that maps the displacement vector to the system outputs. The transfer function of the system

$$\boldsymbol{H}(s) = \boldsymbol{C}\left(s^2\boldsymbol{M}+s\boldsymbol{D}+\boldsymbol{K}\right)^{-1}\boldsymbol{B} \in \mathbb{R}^{k\times l} \tag{4}$$

characterizes the system behavior from the inputs $\boldsymbol{u}$ to the outputs $\boldsymbol{y}$ in the Laplace domain with the Laplace variable $s \in \mathbb{C}$. In the case of $s = i\omega$, $\boldsymbol{H}(i\omega)$ is also called frequency response function.

2.2 Model Order Reduction

A fine spatial finite element discretization is often used to get a good approximation of the continuous system. This causes many degrees of freedom N and high computational cost to solve the equation of motion. To overcome this issue, model order reduction can be applied by approximating the displacement vector with

$$\boldsymbol{q}(t) \approx \boldsymbol{V}\boldsymbol{q}_\mathrm{red}(t), \tag{5}$$

where $\boldsymbol{V} \in \mathbb{R}^{N\times n}$ is the projection matrix and $\boldsymbol{q}_\mathrm{red}(t) \in \mathbb{R}^n$ are the reduced elastic coordinates. Applying an orthogonal Galerkin projection to the equation of motion (3) results in the reduced order equation of motion

$$\begin{aligned} \underbrace{\boldsymbol{V}^\mathsf{T}\boldsymbol{M}\boldsymbol{V}}_{\boldsymbol{M}_\mathrm{red}}\ddot{\boldsymbol{q}}_\mathrm{red}(t)+\underbrace{\boldsymbol{V}^\mathsf{T}\boldsymbol{D}\boldsymbol{V}}_{\boldsymbol{D}_\mathrm{red}}\dot{\boldsymbol{q}}_\mathrm{red}(t)+\underbrace{\boldsymbol{V}^\mathsf{T}\boldsymbol{K}\boldsymbol{V}}_{\boldsymbol{K}_\mathrm{red}}\boldsymbol{q}_\mathrm{red}(t) &= \underbrace{\boldsymbol{V}^\mathsf{T}\boldsymbol{B}}_{\boldsymbol{B}_\mathrm{red}}\boldsymbol{u}(t), \\ \boldsymbol{y}_\mathrm{red}(t) &= \underbrace{\boldsymbol{C}\boldsymbol{V}}_{\boldsymbol{C}_\mathrm{red}}\boldsymbol{q}_\mathrm{red}(t) \end{aligned} \tag{6}$$

with the reduced system matrices $\boldsymbol{M}_{\text{red}}, \boldsymbol{D}_{\text{red}}, \boldsymbol{K}_{\text{red}} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B}_{\text{red}} \in \mathbb{R}^{n\times l}$ and $\boldsymbol{C}_{\text{red}} \in \mathbb{R}^{k\times n}$. Here, $\boldsymbol{y}_{\text{red}}(t) \in \mathbb{R}^k$ is the output of the reduced order system. Model order reduction seeks to find an $n \ll N$ dimensional subspace $\mathscr{V}$ and its corresponding projection matrix $\boldsymbol{V}$ with $\text{colspan}(\boldsymbol{V}) = \mathscr{V}$ such that the dimension of the reduced subspace is as small as possible on the one hand and the error in time domain $|\boldsymbol{y}(t) - \boldsymbol{y}_{\text{red}}(t)|$ and frequency domain $\|\boldsymbol{H}(s) - \boldsymbol{H}_{\text{red}}(s)\|$ is as small as possible on the other hand. Many methods have been developed to find a suitable projection matrix $\boldsymbol{V}$. These methods can be mainly categorized into methods based on modal truncation, methods using singular value decomposition, and methods based on Krylov-subspaces. An overview over different projection-based model order reduction techniques can be found in [3,6].

The method used in this contribution is moment-matching with Krylov-subspaces. It matches the transfer function of the full order and the reduced order model and its derivatives up to the desired i-th order at specific expansion points $\hat{s}_j$ in the Laplace-domain. Therefore, the coefficients, also called moments, of the series expansion of the transfer function have to be calculated. Since the explicit calculation of these moments is numerically unstable, the Arnoldi-algorithm and the Lanczos-algorithm provide a stable and iterative way to match the moments, see e.g. [10,16,17].

In this contribution, systems that depend on a parameter vector $\boldsymbol{p} \in \mathbb{R}^d$ play an important role. The parameter dependency in the system matrices $\boldsymbol{M}(\boldsymbol{p})$, $\boldsymbol{D}(\boldsymbol{p})$ and $\boldsymbol{K}(\boldsymbol{p})$ should also be preserved in the reduced order model. Therefore, it is important that the whole parameter space of interest is also covered in the subspace $\mathscr{V}$. As stated in [7], the global reduction matrix can be obtained by concatenating multiple local projection matrices $\widetilde{\boldsymbol{V}}(\hat{\boldsymbol{p}}_k) \in \mathbb{R}^{N\times n_k}$, which are received by moment-matching at specific parameter points $\hat{\boldsymbol{p}}_k$, to a global reduction matrix $\boldsymbol{V} \in \mathbb{R}^{N\times n}$ with $n = \sum n_k$. A singular value decomposition of $\boldsymbol{V}$ is often applied to get only linear independent columns.

3 Geometrically Parameterized Shell Element Formulation

In [11] a new shape optimization procedure has been derived. This procedure is based on a geometrically parameterized solid finite element that analytically contains the geometry of the element as a parameter. The benefit of this method is, that model order reduction and finite element pre-processing are only done in the offline phase and not

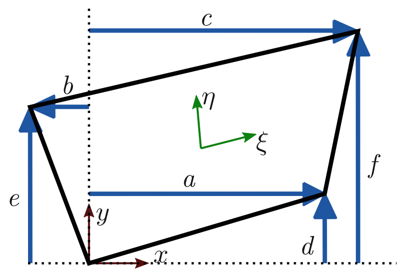


Fig. 1. Node position parameterization with seven local element parameters $\boldsymbol{p}^e = [a\ b\ c\ d\ e\ f\ t]^{\mathsf{T}}$.

repeated in every iteration step in the online phase. A successful application in combination with optimal control can be seen in [12]. Since the procedure is limited to eight node hexahedron elements, this contribution aims to extend the method by a geometrically parameterized finite shell element with four nodes. Figure 1 shows the element parameterization in two-dimensional x-y space. The x-y coordinate system is located in the bottom left node of the element. Hence, there are six parameters left to fully describe the shape of the element, where each parameter represents either the x position or the y position of a node. A seventh parameter is used to model the thickness of the element. The resulting element parameter vector is $\boldsymbol{p}^e = [a\ b\ c\ d\ e\ f\ t]^\mathsf{T}$. Under consideration of the element parameter vector, Eq. (2) also becomes parameter dependent and can be rewritten in an affine form resulting in

$$\boldsymbol{k}^e(\boldsymbol{p}^e) = \int_{-1}^{1}\int_{-1}^{1} \widetilde{\boldsymbol{B}}^\mathsf{T}(\boldsymbol{p}^e)\boldsymbol{H}\widetilde{\boldsymbol{B}}(\boldsymbol{p}^e)\det(\boldsymbol{J}(\boldsymbol{p}^e))\,\mathrm{d}\xi\,\mathrm{d}\eta = \sum_{j=1}^{J} \bar{w}_j^e(\boldsymbol{p}^e)\boldsymbol{k}_j^e \tag{7}$$

with weighting functions $\bar{w}_j^e(\boldsymbol{p}^e) \in \mathbb{R}$ depending on the element parameter vector and constant matrices $\boldsymbol{k}_j^e \in \mathbb{R}^{24\times 24}$. The exact derivation of the parameter dependency is omitted here, but an example derivation for a solid element can be found in [11]. As already mentioned in Sect. 2.1, the results of pure displacement based four node shell elements are not necessarily reliable because of locking or hourglass modes. The methods to overcome these issues yield additional terms in the element stiffness matrix and consequently a higher upper limit of summation J in the affine representation. This results in higher storage cost and higher computational cost for the evaluation of the element stiffness matrix at specific parameter points. It is consequently of great interest that J becomes as small as possible. Therefore, the number of summation terms J is compared for an implementation of an element with full integration and the Enhanced Assumed Strain Method [18] and the Assumed Natural Strain Method proposed in [5]

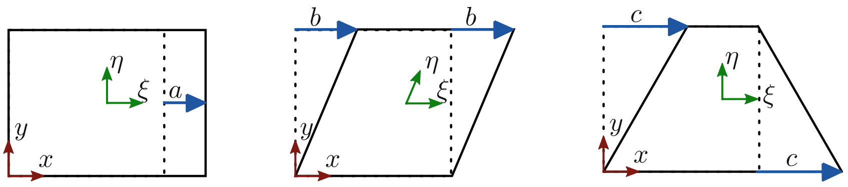


Fig. 2. Mode shape parameterization, where the shape of the element is modeled by a superposition of modes. The parameterization in y direction is similar but tilted by 90°.

Table 1. Number of summation terms J in the element stiffness matrix $\boldsymbol{k}^e(\boldsymbol{p}^e)$ for $\boldsymbol{p}^e = [a\ b\ c\ d\ e\ f\ t]^\mathsf{T}$.

	reduced integration	full integration
node position parameterization	3 203	–
mode shape parameterization	2 814	–

and for an implementation with reduced integration and hourglass-control [14]. The element stiffness matrix does not only depend on the order of Gaussian integration but also on the choice of the parameter vector $\boldsymbol{p}^e$. Another parameterization is shown in Fig. 2. The figure shows three modes, a rectangle mode, a parallelogram mode, and a trapezoid mode. Each of these modes is described by one of the parameters a, b, c. Equivalently, three other parameters d, e, f describe the parameterization in y direction. The shape of the element is a superposition of these six modes. The thickness of the element t completes the list of parameters.

The comparison of these different implementations and parameterizations is done only for the parametric stiffness matrix because the mass matrix is not influenced by anti-locking or anti-hourglassing methods and can easily be calculated by including the parameter vector $\boldsymbol{p}^e$ into Eq. (1) without considering any additional terms. Table 1 shows J for the element stiffness matrix for all combinations. There are no results for full integration, because the computer used for this calculations was not able to calculate the complex matrix inversions symbolically that were necessary for the Enhanced Assumed Strain Method. The upper limit of summation J for both, the parameterization with mode shapes and the parameterization with node positions, for the element with reduced integration are in the same order of magnitude, though J is slightly lower for the mode shape parameterization. To get an idea how complex full integration is compared to reduced integration, the number of summation terms for the parameter vector $\boldsymbol{p}^e = [d\ e\ f\ t]^{\mathsf{T}}$ is shown in Table 2. The element is thus only parameterized in y-direction and it is not possible to yield that many element geometries as with seven element parameters. Obviously, J is $1.5 - 2.2$ times higher for the element with full integration than for the element with reduced integration. This has two reasons. Firstly, the pure displacement based stiffness matrix with reduced integration has a lower number of summation terms than with full integration because of the lower number of integration points. Secondly, the implemented methods to avoid locking induce more additional summation terms than the methods to avoid hourglassing. It can also be stated, that the number of parameters seems to have a major impact on J, since J is a whole order of magnitude lower for $\boldsymbol{p}^e = [d\ e\ f\ t]^{\mathsf{T}}$ than for $\boldsymbol{p}^e = [a\ b\ c\ d\ e\ f\ t]^{\mathsf{T}}$. Nevertheless, the parameterization with seven element parameters is still necessary, because otherwise it would not be possible to represent all possible element geometries.

4 Numerical Example

The previously derived implementation of the parameterized finite element is used in combination with the workflow developed in [11] for a shape optimization example. In this workflow, the geometry of the structure is parameterized by some global design parameters. In the offline phase, a parametric model reduction is performed to reduce the complexity of the system. In the online phase, it is thus only necessary to evaluate the reduced system matrices at specific parameter points and to calculate the cost function. It is therefore not necessary to perform pre-processing steps such as meshing and linear model order reduction in each iteration of the optimization procedure. Additionally, no calculations need to be made with the full order model during the online phase. Depending on the problem definition, these points can be a huge time saver.

4.1 Modeling

The structure used is a plate with two supporting ribs underneath the plate as shown in Fig. 3. For a better visualization, the figure is shown upside down. The size of the base area is 5 m $\times$ 5 m and the structure is mounted at the edges of the slab such that the translational degrees of freedom are locked. As introduced in Sect. 2.1, Rayleigh damping $\boldsymbol{D}(\boldsymbol{p}) = \alpha \boldsymbol{M}(\boldsymbol{p}) + \beta \boldsymbol{K}(\boldsymbol{p})$ is added with $\alpha = 3.5$ and $\beta = 2 \cdot 10^{-5}$. The material of the structure is concrete, the corresponding material properties are chosen from [9] and result in the Young's modulus $E = 30 \cdot 10^9\,{}^{\mathrm{N}}/_{\mathrm{m}^2}$, a Poisson's ratio of $\nu = 0.2$ and a density of $\rho = 2400\,{}^{\mathrm{kg}}/_{\mathrm{m}^3}$. The meshing is done in the commercial finite element software Ansys [2] resulting in $N = 22212$ degrees of freedom. All the other calculations are done in Matlab [15]. The structure is parameterized by four design parameters $\boldsymbol{p} \in \mathbb{R}^4$. The height of the ribs is modelled by a second order Bézier curve. The control points are shown as red dots in Fig. 3. Since the structure is symmetric, all four boundary points are parameterized with the same parameter. This results in two parameters to model the rib height. The third parameter describes the thickness of the plate and the fourth parameter specifies the thickness of the ribs. The global parametric stiffness matrix is constructed by mapping the local element degrees of freedom to the global degrees of freedom using the transformation matrix $\boldsymbol{T}^e \in \mathbb{R}^{N \times 24}$ and yields

$$\begin{aligned} \boldsymbol{K}(\boldsymbol{p}) &= \sum_{e=1}^{n_{\mathrm{Ele}}} \boldsymbol{T}^e \boldsymbol{k}^e(\boldsymbol{p}^e) \boldsymbol{T}^{e\mathsf{T}} = \sum_{e=1}^{n_{\mathrm{Ele}}} \boldsymbol{T}^e \left(\sum_{j=1}^{J} \bar{w}_j^e(\boldsymbol{p}^e) \boldsymbol{k}_j^e \right) \boldsymbol{T}^{e\mathsf{T}} \\ &= \sum_{e=1}^{n_{\mathrm{Ele}}} \boldsymbol{T}^e \left(\sum_{i=1}^{I} \widetilde{w}_i^e(\boldsymbol{p}) \widetilde{\boldsymbol{k}}_i^e \right) \boldsymbol{T}^{e\mathsf{T}} = \sum_{e=1}^{n_{\mathrm{Ele}}} \sum_{i=1}^{I} \widetilde{w}_i^e(\boldsymbol{p}) \boldsymbol{T}^e \widetilde{\boldsymbol{k}}_i^e \boldsymbol{T}^{e\mathsf{T}} = \sum_{l=1}^{L} w_l(\boldsymbol{p}) \boldsymbol{K}_l. \end{aligned} \tag{8}$$

The global stiffness matrix is again in an affine form with weighting functions $w_l(\boldsymbol{p}) \in \mathbb{R}$ depending on the global design vector $\boldsymbol{p}$ and constant matrices $\boldsymbol{K}_l \in \mathbb{R}^{N \times N}$. One important step is after the third equal sign of Eq. (8). The element stiffness matrix is reformulated with $\boldsymbol{p}^e = f^e(\boldsymbol{p})$, such that it no longer depends on the element parameter vector $\boldsymbol{p}^e$ but on the global design parameter vector $\boldsymbol{p}$. The global parametric mass matrix is obtained in the same manner. The number of summation terms L in the global parametric stiffness matrix is bounded by $L \leq n_{\mathrm{Ele}} J$. Due to the fact that often not all local element parameters are used in the global parameterization and some terms can be summarized, L is typically much smaller. Table 3 shows L for all element implementations mentioned in the previous section. This can of course not be generalized on all structures but is valid for this structure with this specific parameterization. One remarkable thing is that mode shape parameterization and the node position parameterization have exactly the same upper limit of summation L although $\boldsymbol{p}^e$ is different

Table 2. Number of summation terms J in the element stiffness matrix $\boldsymbol{k}^e(\boldsymbol{p}^e)$ for $\boldsymbol{p}^e = [d\ e\ f\ t]^{\mathsf{T}}$.

	reduced integration	full integration
node position parameterization	209	471
mode shape parameterization	99	156

Table 3. Number of summation terms L for the global stiffness matrix $\boldsymbol{K}(\boldsymbol{p})$.

	reduced integration	full integration
node position parameterization	2 058	7 621
mode shape parameterization	2 058	7 621

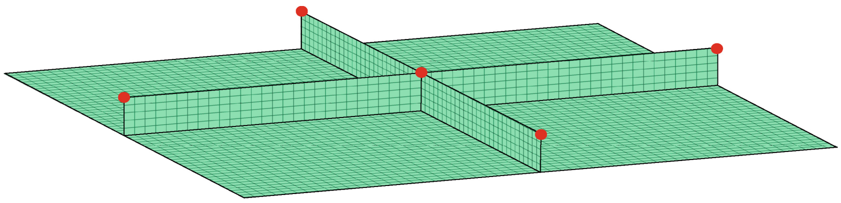


Fig. 3. Initial geometry before the optimization with $\boldsymbol{p} = [0.30\ 0.30\ 0.03\ 0.03]^{\mathsf{T}}$.

for both parameterizations. This is not a coincidence but due to the fact that the same global design parameters $\boldsymbol{p}$ were chosen. In general, the resulting stiffness matrix is only influenced by the global design parameters and the chosen element formulation but not by the element parameterization $\boldsymbol{p}^e$. Thus not only the upper limit of summation is identical for both parameterizations but also the whole stiffness matrix. As expected, the number of summation terms is much higher when elements with full integration are used compared to when reduced integration is used.

To reduce the numerical complexity during the optimization procedure, model order reduction as explained in Eqs. (5) and (6) is applied to the system matrices. Therefore, six linearly distributed expansion points are chosen in the frequency domain between 0 Hz and 100 Hz and 15 randomly selected parameter shifts are selected in the parameter space $\mathscr{P}$. The parameter domain is defined such that the thickness of the ribs and the plate is within 0.01 m and 0.05 m and the rib height is in the range between 0 m and 0.5 m at the control points of the Bézier curve. The resulting reduced order model has $n = 300$ degrees of freedom. Figure 4 shows the Frobenius norm of the frequency response function for the full order model and the reduced order model for a randomly selected point in the parameter domain that was not a part of the 15 points used for reduction. The figure also shows the absolute reduction error $||\boldsymbol{H} - \boldsymbol{H}_{\text{red}}||_{\text{F}}$. The reduced system approximates the full model very well, because the reduction error is about two orders of magnitude lower than the frequency response of the original system.

4.2 Optimization

The obtained reduced order model is able to approximate the full order model in the frequency range from 0 Hz to 100 Hz and in the given parameter domain. For the optimization task examined in this paper a surface load of $1.8 \cdot 10^3\,\mathrm{N/m^2}$ is applied perpendicular to the plate surface. The cost function and the suitable parameter space is formulated as

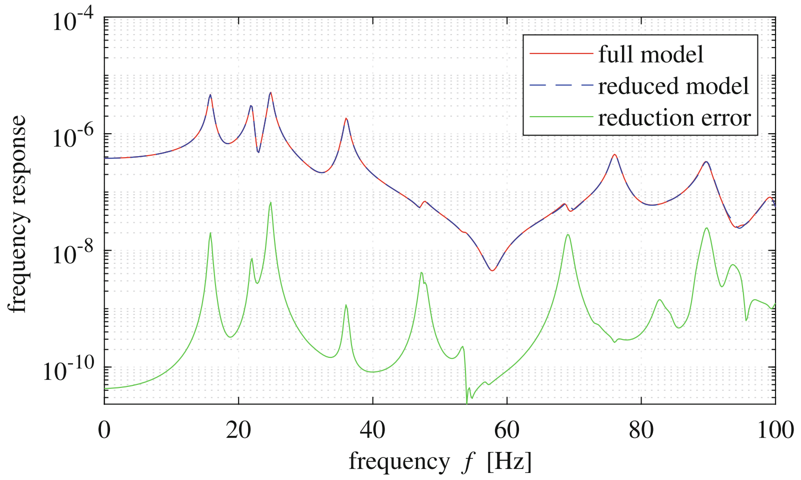


Fig. 4. Frequency response function and reduction error for a randomly selected parameter point $\hat{\boldsymbol{p}}_{\text{test}} = [0.48\ 0.23\ 0.04\ 0.02]^{\mathsf{T}}$ in the parameter space $\mathscr{P}$.

$$\begin{aligned}
\boldsymbol{p}^* &= \underset{\boldsymbol{p}\in\mathscr{P}}{\arg\min}\, J(\boldsymbol{p}) = m(\boldsymbol{p}) = \rho\left(25p_3 + \frac{20}{3}p_1p_4 + \frac{10}{3}p_2p_4\right) \\
\text{s.t.} \quad & \mathscr{P} := \left\{\boldsymbol{p}\in\mathbb{R}^4 \,|\, \boldsymbol{h}(\boldsymbol{p}) \le \boldsymbol{0} \,|\, \boldsymbol{p}_{\text{low}} \le \boldsymbol{p} \le \boldsymbol{p}_{\text{up}}\right\} \\
& h_1(\boldsymbol{p}) = p_1 + p_2 - 1 \le 0, \quad h_2(\boldsymbol{p}) = -p_1 - p_2 \le 0, \quad h_3(\boldsymbol{p}) = \max\left(\boldsymbol{K}^{-1}\boldsymbol{B}\boldsymbol{u}\right) - 0.01 \le 0 \\
& \boldsymbol{p}_{\text{low}} = [0\ -0.5\ 0.01\ 0.01]^{\mathsf{T}}, \qquad \boldsymbol{p}_{\text{up}} = [0.5\ 1\ 0.05\ 0.05]^{\mathsf{T}}.
\end{aligned} \tag{9}$$

The objective is to minimize the mass of the structure. Additionally to the parameter space defined before, the maximum deflection perpendicular to the plate surface is limited to 0.01 m by $h_3(\boldsymbol{p})$ during the optimization. This allows to find a structure that is as light as possible while the deformations are bounded within the given tolerance. The search for potential savings in material use is an important research question in the Collaborative Research Centre 1244 (CRC 1244) "Adaptive Skins and Structures for the Built Environment of Tomorrow" at the University of Stuttgart, where this project is based. The optimization problem does not include the mass matrix, i.e. it only considers the steady state deflection after a static load case. As can be seen in Fig. 4, the reduced order model is also able to approximate static loads, i.e. loads at $f = 0\,\text{Hz}$.

The optimization is done with the Matlab algorithm *fmincon*. The resulting geometry is depicted in Fig. 5. The thickness of the ribs is at the lower bound since its contribution to the stiffness of the system perpendicular to the plate surface is very low. The ribs are of parabolic shape with the height at the middle control point at the upper limit of 0.5 m. Without the ribs, the largest deflection would be in the middle of the slab, therefore the support by the ribs is highest there.

4.3 Validation

In this section, the parameterization of the finite element is validated using the results from the optimization. Therefore, the node positions of the optimized geometry are exported to Ansys, where the system matrices are calculated. After transferring them back to Matlab, the behavior of the system obtained with the parameterized reduced oder model can be compared to the behavior of the Ansys model. Since the validation of the reduction process has already been done in Fig. 4, this section uses the full order parametric model to compare it with the results from Ansys. The Frobenius norm of the frequency response function for both models and the corresponding absolute error is shown in Fig. 6. Only at the eigenfrequencies, the frequency response error becomes larger. This could be caused by different correction factors for the hourglass stabilization in Ansys in contrast to the Matlab implementation or further technologies in Ansys, e.g., for distorted meshes, which are not taken into account in the Matlab implementation.

To further validate the implementation, mode shapes of both systems are compared with the Modal Assurance Criterion, see [1], that compares the i-th eigenvector of the Ansys model with the j-th eigenvector of the reduced order Matlab model. It evaluates the similarity on a scale from 0 to 1, where 1 means that the modes are completely identical. The result is given in Fig. 7. The diagonal entries are close to one, which means a good agreement of the eigenmodes. Only the second and the third eigenmodes do not correspond that well. Due to the symmetry of the structure, the second and third eigenmode have the same eigenfrequency. With double eigenfrequencies, all possible

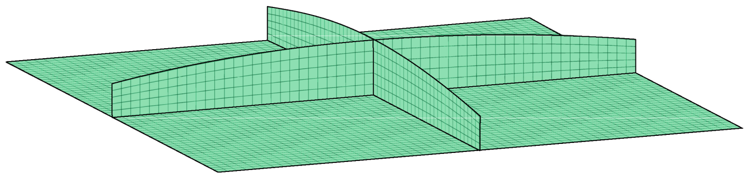


Fig. 5. Optimized geometry with $\boldsymbol{p}^* = [0.30\ 0.70\ 0.02\ 0.01]^\mathsf{T}$.

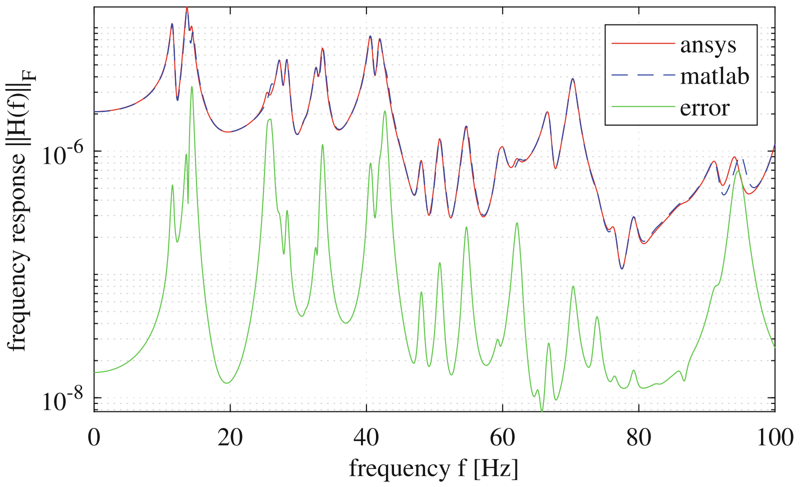


Fig. 6. Frequency response function for the optimized geometry $\boldsymbol{p} = \boldsymbol{p}^*$.

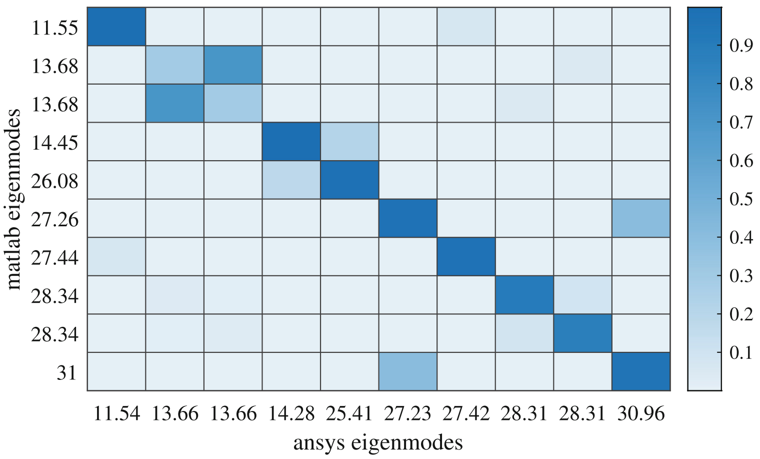


Fig. 7. Modal Assurance Criterion and eigenfrequencies for the optimized geometry $\boldsymbol{p} = \boldsymbol{p}^*$ to compare the results with results from Ansys.

linear combinations of the two eigenvectors are also potential eigenvectors. The second and third eigenmode of the Matlab model are only another linear combination and are thus valid. The axis labels of the figure als shows the eigenfrequencies of both models. Generally, the relative error of the eigenfrequencies is far below 1 %, only the fifth eigenfrequency is around 2 %. Summarizing the results from Fig. 6 and 7, with the parameterized model it is possible to obtain good results also for a distorted mesh of the structure in terms of the frequency response, the mode shapes and the eigenfrequencies.

5 Conclusion and Outlook

In this contribution, the workflow for shape optimization with geometrically parameterized finite elements developed in [11] was extended by a parameterized shell element. To get an optimal implementation regarding the number of summation terms, different element technologies were compared. The most efficient implementation was an element with reduced integration and hourglass control. Afterwards, the implemented shell element was used for a shape optimization example in combination with parametric model order reduction. A validation with the system matrices obtained from Ansys is done using the frequency response function and the Modal Assurance Criterion. A good agreement between the two models was found.

Since the number of summation terms is very high for the element implementation and this also influences the number of summation terms in the global system, an interesting research topic for the future is to further reduce the number of summation terms by implementing series expansions or low rank approximations. Furthermore, it is not clear that two supporting ribs in the middle of the plate, as examined in this contribution, are the best topology for this optimization problem. A future work could also include coupled parameterized systems, where the number and position of the ribs, or more generally the coupling of two or more bodies is included in the optimization problem.

Acknowledgements. This project is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - Project-ID 279064222 - SFB 1244, “Adaptive skins and structures for the built environment of tomorrow” with the project B01 (Characterization, modeling and model order reduction). This support is highly appreciated.

References

1. Allemang, R.: The modal assurance criterion - 20 years of use and abuse. In Proceedings of the 20th International Modal Analysis Conference, Los Angeles, USA, pp. 14–21 (2002)
2. Ansys: Documentation for Ansys, Release 18.2. Ansys, Inc. (2018)
3. Antoulas, A.: Approximation of Large-Scale Dynamical Systems. SIAM, Philadelphia (2005)
4. Bathe, K.J.: Finite Element Procedures. Prentice Hall, Upper Saddle River (2014)
5. Bathe, K.J., Dvorkin, E.N.: A four-node plate bending element based on Mindlin/Reissner plate theory and a mixed interpolation. Int. J. Numer. Meth. Eng. **21**(2), 367–383 (1985)
6. Benner, P., Ohlberger, M., Cohen, A., Willcox, K.: Model Reduction and Approximation. SIAM, Theory and Algorithms (2017)
7. Benner, P., Gugercin, S., Willcox, K.: A survey of projection-based model reduction methods for parametric dynamical systems. SIAM Rev. **57**(4), 483–531 (2015)
8. Neto, M.A., Amaro, A., Roseiro, L., Cirne, J., Leal, R.: Finite element method for plates/shells. In: Engineering Computation of Structures: The Finite Element Method, pp. 195–232. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-17710-6_6
9. DIN Standards Commitee Building and Civil Engineering: In Eurocode 2: Design of Concrete Structures - Part 1–1: General Rules - Rules for Buildings. Beuth Verlag, Bridges and Civil Engineering Structures; Berlin (2021)
10. Fehr, J.: Automated and Error-Controlled Model Reduction in Elastic Multibody Systems. Dissertation, Schriften aus dem Institut für Technische und Numerische Mechanik der Universität Stuttgart, Vol. 21. Shaker Verlag, Aachen (2011)
11. Fröhlich, B., Gade, J., Geiger, F., Bischoff, M., Eberhard, P.: Geometric element parameterization and parametric model order reduction in finite element based shape optimization. Comput. Mech. **63**(5), 853–868 (2019)
12. Fröhlich, B.; Wagner, J.; Böhm, M.; Sawodny, O.; Eberhard, P.: Combining Optimal control and shape optimization for an adaptive engineering structure with parameterized reduced order finite element models. In: Proceedings of the 9th ECCOMAS Thematic Conference on Smart Structures and Materials, Paris, France, pp. 43–54 (2019)
13. Ibrahimbegovic, A., Taylor, R.L., Wilson, E.L.: A robust quadrilateral membrane finite element with drilling degrees of freedom. Int. J. Numer. Meth. Eng. **30**(3), 445–457 (1990)
14. Liu, W.K., Ong, J.S.J., Uras, R.A.: Finite element stabilization matrices, a unification approach. Comput. Methods Appli. Mech. Eng. **53**(1), 13–46 (1985)
15. Mathworks: Matlab, Product Help, Matlab Release 2020b. The MathWorks, Inc. (2020)
16. Panzer, H.K.F.: Model Order Reduction by Krylov Subspace Methods with Global Error Bounds and Automatic Choice of Parameters. Dissertation, Technische Universität München. München: Verlag Dr. Hut (2014)
17. Salimbahrami, S.B.: Structure Preserving Order Reduction of Large Scale Second Order Models. Dissertation, Technische Universität München. München (2005)

18. Simo, J.C., Rifai, M.S.: A class of mixed assumed strain methods and the method of incompatible modes. Int. J. Numer. Meth. Eng. **29**(8), 1595–1638 (1990)
19. Zienkiewicz, O.C., Taylor, R.L.: The Finite Element Method for Solid and Structural Mechanics. Butterworth-Heinemann, Oxford (2005)
20. Zienkiewicz, O.C., Taylor, R.L., Zhu, J.Z.: The Finite Element Method - Its Basis & Fundamentals. Butterworth-Heinemann, Oxford (2005)

Efficient Robust Topology Optimization of Eigenfrequencies Using the First-Order Second-Moment Method

Jan Christoph Krüger(✉) and Benedikt Kriegesmann

Structural Mechanics for Lightweight Design, Hamburg University of Technology, Hamburg, Germany
{jan.krueger,benedikt.kriegesmann}@tuhh.de

1 Introduction

Design optimization has become an important field in engineering science since it allows a significant reduction in mass, cost and environmental impact. It can be divided in several fields such as parameter optimization, shape optimization and topology optimization. A typical objective in optimization of dynamic systems is the lowest eigenfrequency of a component. There are several challenges like repeated eigenfrequencies, localized modes in low density areas and high computational effort which have been solved by several scientists (see e.g. [1,5,14]). Deterministic optimization however sometimes leads to designs that are sensitive to deviations from the ideal configuration (eg. load direction or shape).

Various classes of methods have been developed in order to get optimized designs that are not sensitive to imperfections, such as robust design optimization (RDO) and reliability-based design optimization (RBDO). An overview is given in [19]. RBDO usually reduces the risk of failure by dividing the random space into a fail region and a safe region. For calculating the probability of failure, an internal optimization problem has to be solved. This usually results in high computational cost. RDO instead minimizes the variability of the objective function. In the current paper we focus on probabilistic methods and the minimization of mean and standard deviation of the objective function.

Popular probabilistic methods are the Monte-Carlo method (see e.g. [8]) and Taylor-series-based methods. The Monte-Carlo method is very accurate, but it requires a large amount of function evaluations. For this reason, the objective is often approximated by surrogate models such as neural networks [4]. However, the costly training of the surrogate models must be done in each optimization step again.

In contrast to the Monte-Carlo method, Taylor-series-based methods approximate functions by a Taylor-series. Using this approximation, analytic formulas for calculating mean and standard deviation are obtained. One example is the perturbation approach [10], which is based on a Taylor expansion of the static equilibrium. The computational cost here scales with the number of random parameters. In difference to that, the first-order second-moment method (FOSM) is based on an expansion of the objective function itself. Embedding it into robust design optimization for minimizing compliance yields an approach, which requires only the solution of one additional adjoint system

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 75–84, 2024.
https://doi.org/10.1007/978-3-031-50000-8_7

per iteration [9]. Hence, the computational cost is only increased by factor 2 (or less) compared to a deterministic optimization. Because both methods are based on a Taylor series, the accuracy is comparable (given the same approximation order) and much lower than the accuracy of the Monte-Carlo method. For this reason the current paper uses FOSM during the optimization and Monte-Carlo as a benchmark in the analysis of the results.

Due to the large amount of design parameters in topology optimization only gradient-based optimization can be used [16]. For this reason, the gradients of the robust objective have to be computed. In the current paper, a new approach for calculating the gradient of a robust objective using the first-order second-moment method is provided. It only doubles the computational cost compared to a deterministic optimization and can be used independently of the problem-type. This paper is organized as follows: First, the general probabilistic framework is presented, afterwards, the special problem of optimizing eigenfrequencies is described. This is followed by different ways to calculate the gradients of the robust objective. The theoretical concepts are shown for a numerical example.

2 Robust Topology Optimization of Eigenfrequencies

The general procedure for a robust design optimization is similar to a deterministic optimization. However mean value μ_f and standard deviation σ_f are minimized at the same time, which yields a multi-objective optimization problem. This is reduced to a single objective by

$$f_p(\boldsymbol{y},\boldsymbol{x}) = \mu_f(\boldsymbol{y},\boldsymbol{x}) + \kappa\sigma_f(\boldsymbol{y},\boldsymbol{x}) \tag{1}$$

with $\boldsymbol{x}$ as random parameters, $\boldsymbol{y}$ as design parameters and κ as a chosen weighting factor.

2.1 Computation of Eigenfrequencies and Its Gradients

The current paper addresses a robust optimization of the lowest eigenfrequency. Eigenfrequencies of an undamped FE-model are calculated by the solution of an eigenvalue problem given by

$$\left[\boldsymbol{K} - \omega^2\boldsymbol{M}\right]\boldsymbol{\varphi} = \boldsymbol{0} \tag{2}$$

with the eigenvalue ω^2, eigenvector $\boldsymbol{\varphi}$, stiffness matrix $\boldsymbol{K}$ and mass matrix $\boldsymbol{M}$. The eigenfrequency ω is computed as a square root of the eigenvalue ω^2. In the following optimizations, the squared eigenfrequency ω^2 is considered as objective function. According to [6], first-order sensitivities of ω^2 are computed by

$$\frac{\mathrm{d}\omega^2}{\mathrm{d}x_i} = \boldsymbol{\varphi}^T\left[\frac{\mathrm{d}\boldsymbol{K}}{\mathrm{d}x_i} - \omega^2\frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}x_i}\right]\boldsymbol{\varphi} \quad . \tag{3}$$

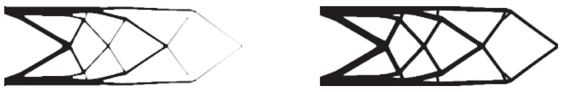
In order to comply with the convention to minimize an objective function, the deterministic objective equals the negative eigenfrequency $f = -\omega^2$. Realistic problems have repeated and nearby eigenvalues. When only the lowest eigenvalue is considered, mode switches might occur during the optimization. In order to avoid this problem, the m smallest eigenfrequencies are aggregated using the p-norm (see for example [5]).

2.2 Robust Topology Optimization Framework

The optimization is parametrized using the SIMP-based topology optimization scheme [2]. In order to overcome numerical instabilities, a variable filter [18] with filter padding [3] is used. Without further measures, this would end up in "grey" designs. To overcome this problem, the filtered parameters are projected using a heavyside-like function in the form given in [18] as

$$\rho_i = \frac{\tanh(\beta_i \eta_i) + \tanh\left(\beta\left[\hat{y}_i - \eta_i\right]\right)}{\tanh(\beta \eta_i) + \tanh\left(\beta\left[1 - \eta_i\right]\right)} \tag{4}$$

with the projection parameter β, threshold parameter η_i and filtered variable $\hat{y}_i$ for element i. In order to model geometric imperfections the threshold parameter is scattered like first time seen in [15]. In that paper, the projection uses a global parameter η. If this value increases, the whole model is eroded, if η decreases, the whole model is dilated like shown in Fig. 1. In the current paper, the threshold parameter is defined individually for each element like done in [13]. With this definition, local erosion and dilation can be modelled. In order to model correlation between neighbour elements, isotropic Gauß random fields are used.



(a) Inceased $\eta \rightarrow$ eroded design (b) Decreased $\eta \rightarrow$ dilated design

Fig. 1. Influence of changed threshold parameter η: higher η causes erosion, lower η causes dilation.

As shown in Eq. (1), the robust objective consists of stochastic moments of the deterministic objective. The general framework is taken from [9]. Using the first-order second-moment method, mean and variance of an objective f are computed by

$$\mu_f = f(\mu_x) \tag{5}$$

$$\sigma_f^2 = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}^T (\mu_x) \cdot \boldsymbol{Cov} \cdot \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} (\mu_x) \quad . \tag{6}$$

The standard deviation is computed by $\sigma_f = \sqrt{\sigma_f^2}$. These formulas contain the objective gradient $\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}$ with respect to random variables $\boldsymbol{x}$ and the covariance matrix $\boldsymbol{Cov}$. First-order sensitivities of mean and variance with respect to design variables $\boldsymbol{y}$ are given by

$$\frac{\mathrm{d}\mu_f}{\mathrm{d}\boldsymbol{y}} = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}(\mu_x) \quad , \tag{7}$$

$$\frac{\mathrm{d}\sigma_f^2}{\mathrm{d}\boldsymbol{y}} = 2 \cdot \frac{\mathrm{d}^2 f}{\mathrm{d}\boldsymbol{y}\mathrm{d}\boldsymbol{x}}(\mu_x) \cdot \boldsymbol{Cov} \cdot \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}(\mu_x) \quad . \tag{8}$$

Here $\frac{\mathrm{d}^2 f}{\mathrm{d}\boldsymbol{y}\mathrm{d}\boldsymbol{x}}$ describes the second-order sensitivity of the objective function with respect to random variables $\boldsymbol{x}$ and design variables $\boldsymbol{y}$. The standard deviation gradient is computed from the variance gradient. In the overall framework, only f, $\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}$, $\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}$ and $\frac{\mathrm{d}^2 f}{\mathrm{d}\boldsymbol{y}\mathrm{d}\boldsymbol{x}}$ have to be computed. The first two terms are known from a deterministic optimization. $\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}$ is computed the same way as $\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}$. Therefore the computational cost is identical to a deterministic optimization. However, the gradient evaluation of the variance is computational costly due to the second-order derivative $\frac{\mathrm{d}^2 f}{\mathrm{d}\boldsymbol{y}d\boldsymbol{x}}$. In order to avoid this problem, there are different approaches to compute the gradient of the variance directly.

3 Computation of the Gradient of the Variance Using the First-Order Second-Moment Method

Previously the first-order second-moment method has been shown for an arbitrary objective. However the gradient of the variance is very hard to compute. Therefore different methods are presented and compared for the eigenfrequency example.

3.1 Direct Differentiation

In the direct differentiation one computes the mixed partial derivative $\frac{\mathrm{d}^2\omega^2}{\mathrm{d}\boldsymbol{y}d\boldsymbol{x}}$ directly and afterwards uses Eq. (8) to compute the variance gradient. Direct differentiation of Eq. (3) leads to

$$\frac{\mathrm{d}^2\omega^2}{\mathrm{d}y_j\mathrm{d}x_i} = 2\boldsymbol{\varphi}^T\left[\frac{\mathrm{d}\boldsymbol{K}}{\mathrm{d}x_i} - \omega^2\frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}x_i}\right]\frac{\mathrm{d}\boldsymbol{\varphi}}{\mathrm{d}y_j} + \boldsymbol{\varphi}^T\left[\frac{\mathrm{d}^2\boldsymbol{K}}{\mathrm{d}y_j\mathrm{d}x_i} - \omega^2\frac{\mathrm{d}^2\boldsymbol{M}}{\mathrm{d}y_j\mathrm{d}x_i} - \frac{\mathrm{d}\omega^2}{\mathrm{d}y_j}\frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}x_i}\right]\boldsymbol{\varphi} \quad . \quad (9)$$

The eigenvector derivative $\frac{\mathrm{d}\boldsymbol{\varphi}}{\mathrm{d}y_k}$ is computed by solution of

$$\left[\boldsymbol{K} - \omega^2\boldsymbol{M}\right]\frac{\mathrm{d}\boldsymbol{\varphi}}{\mathrm{d}y_i} = -\left[\frac{\mathrm{d}\boldsymbol{K}}{\mathrm{d}y_i} - \omega^2\frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}y_i} - \frac{\mathrm{d}\omega^2}{\mathrm{d}y_i}\boldsymbol{M}\right]\boldsymbol{\varphi} \quad . \quad (10)$$

This equation is a singular system of equations. The solution is found using Nelsons method [11]. An overall formula of variance gradient is given by

$$\begin{aligned}\frac{\mathrm{d}\sigma^2_{\omega^2}}{\mathrm{d}y_k} = \sum_i\sum_j\Bigg\{&\Bigg[2\boldsymbol{\varphi}^T\left(\frac{\mathrm{d}\boldsymbol{K}}{\mathrm{d}x_i} - \omega^2\frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}x_i}\right)\frac{\mathrm{d}\boldsymbol{\varphi}}{\mathrm{d}y_k}\\ &+ \boldsymbol{\varphi}^T\left(\frac{\mathrm{d}^2\boldsymbol{K}}{\mathrm{d}y_k\mathrm{d}x_i} - \omega^2\frac{\mathrm{d}^2\boldsymbol{M}}{\mathrm{d}y_k\mathrm{d}x_i} - \frac{\mathrm{d}\omega^2}{\mathrm{d}y_k}\frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}x_i}\right)\boldsymbol{\varphi}\Bigg] \qquad (11)\\ &\cdot Cov(x_i, x_j)\cdot\frac{\mathrm{d}\omega^2}{\mathrm{d}x_j}\Bigg\} \quad .\end{aligned}$$

This method has several drawbacks. For every design variable y_i, one has to determine the eigenvector derivative. This ends up in high computational effort for an increasing number of design variables. In order to reduce computational cost, one can use a matrix decomposition for $\left[\boldsymbol{K} - \omega^2\boldsymbol{M}\right]$. This however increases the storage demands significantly. Another issue of this method is the highly intrusive behaviour. In order to compute the requested values source-code access is required. Therefore direct differentiation can only be used in in-house or open source tools.

3.2 Adjoint Method

As an alternative to direct differentiation, the adjoint method is typically used for calculating a gradient. The variance from Eq. (6) is expanded to

$$\sigma_{\omega^2}^2 = \frac{\mathrm{d}\omega^2}{\mathrm{d}\boldsymbol{x}}^T \boldsymbol{Cov} \frac{\mathrm{d}\omega^2}{\mathrm{d}\boldsymbol{x}} - \boldsymbol{\lambda}^T \left[\boldsymbol{K} - \omega^2 \boldsymbol{M}\right] \boldsymbol{\varphi} \tag{12}$$

Differentiation and reordering leads to

$$\begin{aligned}\frac{\mathrm{d}\sigma_{\omega^2}^2}{\mathrm{d}y_k} &= \boldsymbol{\lambda}^T \left[\frac{\mathrm{d}\boldsymbol{K}}{\mathrm{d}y_i} - \omega^2 \frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}y_i} - \frac{\mathrm{d}\omega^2}{\mathrm{d}y_i}\boldsymbol{M}\right] \boldsymbol{\varphi} \\ &+ 2\sum_i \sum_j \boldsymbol{\varphi}^T \left[\frac{\mathrm{d}^2\boldsymbol{K}}{\mathrm{d}x_i \mathrm{d}y_k} - \omega^2 \frac{\mathrm{d}^2\boldsymbol{M}}{\mathrm{d}x_i dy_k} - \frac{\mathrm{d}\omega^2}{\mathrm{d}y_k}\frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}x_i}\right] \boldsymbol{\varphi} \cdot Cov(x_i, x_j) \frac{\mathrm{d}\omega^2}{\mathrm{d}x_j}\end{aligned} \tag{13}$$

with the adjoint system

$$\mathbf{0} = \boldsymbol{\lambda}^T \left[\boldsymbol{K} - \omega^2 \boldsymbol{M}\right] + \boldsymbol{\varphi}^T \cdot 4 \sum_i \sum_j \left[\frac{\mathrm{d}\boldsymbol{K}}{\mathrm{d}x_i} - \omega^2 \frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}x_i}\right] Cov(x_i, x_j) \frac{\mathrm{d}\omega^2}{\mathrm{d}x_j} \quad . \tag{14}$$

The adjoint system Eq. (14) again is singular. In difference to the eigenvector derivative for direct differentiation this problem can not be solved using Nelsons approach. The image of $\left[\boldsymbol{K} - \omega^2 \boldsymbol{M}\right]$ does not include the corresponding eigenvector. However the right hand side is not orthogonal to the considered eigenvector. Therefore this system of equations has no solution.

3.3 Finite Differences

Different to the adjoint method and direct differentiation, finite differences do not need any internal model information. Only objective function and gradient with respect to design variables are required. The FOSM terms $\frac{\mathrm{d}\omega^2}{\mathrm{d}\boldsymbol{x}}$ and $\frac{\mathrm{d}^2\omega^2}{\mathrm{d}\boldsymbol{x}d\boldsymbol{y}}$ are approximated using finite differences. This method is used in [17] for a robust sizing optimization with FOSM in Abaqus. However finite differences require one function (and gradient) evaluation per random variable. This ends up in a huge computational cost for large numbers of design variables.

3.4 Principal Sensitivity FOSM

As a different method, a new approach called "principal sensitivity FOSM" (PSF) is presented for an arbitrary objective function f. Consider a Taylor series of the objective gradient $\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}$ as a function of random parameters $\boldsymbol{x}$ given by

$$\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}(\mu_x + \Delta\boldsymbol{x}) = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}(\mu_x) + \frac{\mathrm{d}^2 f}{\mathrm{d}\boldsymbol{y}\mathrm{d}\boldsymbol{x}}(\mu_x) \cdot \Delta\boldsymbol{x} + HOT \tag{15}$$

with the higher order terms HOT. We now define the "principal sensitivity" direction $\boldsymbol{Cov} \cdot \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}(\mu_x)$ and choose $\Delta \boldsymbol{x}$ as

$$\Delta \boldsymbol{x} = \varepsilon \cdot \boldsymbol{Cov} \cdot \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}(\mu_x) \quad . \tag{16}$$

The principal sensitivity direction is similar to the projection direction in [7]. After some reordering, the Taylor series has the form

$$\varepsilon \cdot \frac{\mathrm{d}^2 f}{\mathrm{d}\boldsymbol{y} d\boldsymbol{x}}(\mu_x) \cdot \boldsymbol{Cov} \cdot \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}(\mu_x) = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}(\mu_x + \varepsilon \cdot \boldsymbol{Cov} \cdot \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}(\mu_x)) - \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}(\mu_x) + HOT \quad . \tag{17}$$

which is similar to Eq. (8). The only difference is a scaling factor $\frac{\varepsilon}{2}$ and HOT. Neglecting higher order terms, one gets the approximation

$$\frac{\mathrm{d}\sigma^2}{\mathrm{d}\boldsymbol{y}} \approx \frac{2}{\varepsilon}\left[\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}(\mu_x + \varepsilon \cdot \boldsymbol{Cov} \cdot \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}(\mu_x)) - \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{y}}(\mu_x)\right] \quad . \tag{18}$$

The scaling factor ε should be chosen small in order to reduce influence of higher order terms. Using this method, one does not need any modifications in the source code of some FE-software as long as the first-order sensitivities with respect to the design variables and random variables are given. In that case only two objective gradient evaluations are required. Therefore, the computational effort is only doubled compared to a deterministic optimization. Storage requirements are not extended significantly since only one gradient has to be stored additionally.

3.5 Comparison of Methods

Different methods for computing variance gradient with FOSM have been derived. The methods are separated into two groups. Direct differentiation and the adjoint method are analytic methods and require access to the FE-Software source code in order to get the required (second-order) matrix derivatives. In difference finite differences and principal sensitivity FOSM only require gradients of the objective function. In commercial software, the gradients for many combinations of objective functions and design parameters are given. Hence, this class of approaches can be implemented as a plugn to existing codes.

A big difference between the methods is the computation time. Finite differences and direct differentiations need to solve one equation system for each random variable. In difference to that, adjoint method and principal sensitivity FOSM only require one extra solution of a problem. However, it turned out that adjoint method does not work for frequency-response problems. The computation time for the cantilever example considered in Sect. 4 is shown for different methods in Fig. 2. In the current implementation, finite differences and direct differentiation have a complexity order of about 2.1 while PSF only increases linearly. At a number of 250 elements, PSF becomes the most efficient method. At 10.000 elements there is a difference between direct differentiation and PSF of factor 80.

Since PSF is a numeric approximation of the variance gradient, one has to investigate the accuracy. In general the derivation of PSF is analogue to finite differences.

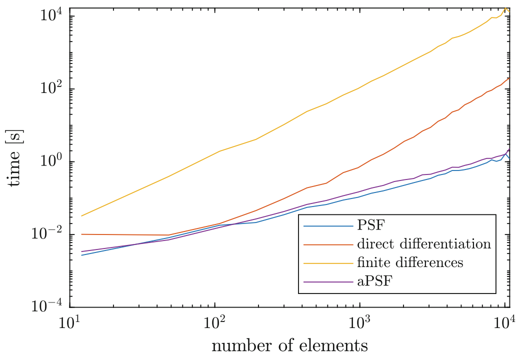


Fig. 2. Required time for FOSM-gradient determination of later example using direct differentiation and principal sensitivity FOSM at different discretization. There is one random and design variable per element.

For this reason, it is expected that the accuracy will be the same. In Fig. 3, one can see the relative error of PSF compared to the analytic direct differentiation solution at a step size $\varepsilon \cdot \left\| \boldsymbol{Cov}\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} \right\|$ of 10^{-5}. Relative means that the error of every entry is divided by the corresponding correct value. The plot shows that the mean error is less than 1% and the median even below 10^{-5}. However, the error of some entries is quite high at about 100%. Further investigation shows that the high relative errors occur at entries which are near to zero and hence, the gradient direction is not significantly influenced. Numerical studies for the example considered in the next section showed that the best accuracy is obtained with a step size of 10^{-5}.

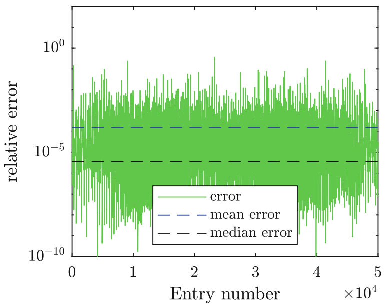
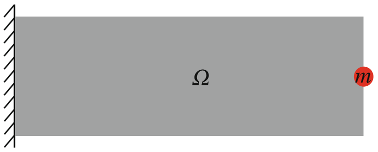


Fig. 3. Relative accuracy of principal sensitivity FOSM for the optimized design of a cantilever beam. Relative means each entry of error vector compared to correct entry.

4 Cantilever Beam Example

The previously derived PSF method is used for eigenfrequency optimization of a cantilever beam like presented in [5] with a design space Ω of 130×390 elements. The model and its properties are shown in Fig. 4. Random geometry is modelled by a Gauß random field with scattering η-values. The optimization is done using a projected gradient descent algorithm ([12]) and a minimal step-size of 10^{-5} as convergence criterium. In order to end up in a better local minimum, the projection parameter β is increased from one to ten like in [9, 13]. The optimization starts from a grey design with a uniform density $\boldsymbol{\rho}$ of 0.3.

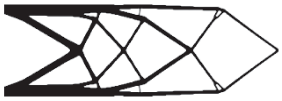


Design Space	130×390
element size	1×1
density, modulus	1
poisson ratio	0.3
SIMP exponent	3
filter radius	3
projection β	increasing $1-10$
mass m at end	3150
maximum volume	12675
modulus empty element E_0	10^{-3}
weighting factor κ	20
standard deviation of η	0.05
correlation length l	30

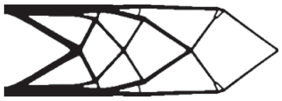
Fig. 4. Model and properties of the cantilever example.

It turns out that a robust design optimization ends up in a very bad local minimum since an increasing β changes the robust objective significantly. Performing a robust design optimization without increasing β leads to bad designs too, because the robust objective has lots of local minima near to the starting design. In order to get good robust designs, the robust optimization is started from a deterministic optimized design. The optimized designs are tested using a Monte-Carlo simulation with a sample size of 12000.

The results are summarized in Fig. 5. The designs significantly differ from the design given in [5]. This is due to a different optimization algorithm and differently chosen filter parameters. The robust and the non-robust design are similar. However one can see that the thin bars are slightly dilated while thick bars are little eroded for the robust design. The small differences result in a 30% reduced standard deviation at the cost of a 0.9% decreased mean value. One can observe that FOSM underestimates the variance by 35%. This is due to nonlinear effects. The comparison of the Monte-Carlo results for both designs reveal that despite this inaccuracy the RDO using FOSM provided a more robust design.



Deterministic optimized design.



Robust optimized design using FOSM.

	Optimization method	
	deterministic	PSF
mean - FOSM	695	689
mean - MC	687	683
standard deviation - FOSM	6.0	4.1
standard deviation - MC	8.8	6.3

Fig. 5. Optimized designs after different optimizations and corresponding function values evaluated with FOSM and Monte Carlo at a sample size of 12000.

5 Conclusions

Different approaches for robust design optimization using the first-order second-moment method are presented. The key problem is the determination of the variance gradient. The adjoint method does not work for eigenfrequencies while direct differentiation and finite differences are computational costly. As an alternative, a new method called principal sensitivity FOSM for gradient computation is presented. It only requires one additional objective gradient evaluation independent of the number of design variables or random variables. In the consequence, it is one complexity order faster than direct differentiation and finite differences. At the same time, it only requires as much storage as a deterministic optimization and can be implemented without access to the source code of FE-software. As a drawback, it is hard to find an appropriate step size for high accuracy results without significant rounding errors. Using the provided method, a robust design optimization of a cantilever is done. Compared to a deterministic optimized design, the standard deviation is reduced by 30% at the cost of less then 1% mean. Future work addresses the problem of rounding errors at small step-sizes and big models.

Acknowledgements. Financial support of the German Research Foundation (reference number KR 4914/13-1) is acknowledged.

References

1. Achtziger, W., Kočvara, M.: On the maximization of the fundamental eigenvalue in topology optimization. Struct. Multidiscip. Optim. **34**(3), 181–195 (2007). https://doi.org/10.1007/s00158-007-0117-3
2. Bendsøe, M.P., Sigmund, O.: Topology Optimization Theory, Methods, and Applications. Springer-Verlag, Berlin (2004). https://doi.org/10.1007/978-3-662-05086-6
3. Clausen, A., Andreassen, E.: On filter boundary conditions in topology optimization. Struct. Multidiscip. Optim. **56**(5), 1147–1155 (2017). https://doi.org/10.1007/s00158-017-1709-1
4. Dey, S., et al.: Uncertain natural frequency analysis of composite plates including effect of noise - A polynomial neural network approach. Compos. Struct. **143**, 130–142 (2016). https://doi.org/10.1016/j.compstruct.2016.02.007

5. Ferrari, F., Lazarov, B.S., Sigmund, O.: Eigenvalue topology optimization via efficient multilevel solution of the frequency response. Int. J. Numer. Meth. Eng. **115**, 872–892 (2018). https://doi.org/10.1002/nme.5829
6. Fox, R.L., Kapoor, M.P.: Rates of change of eigenvalues and eigenvectors. AIAA J. **6**(12), 2426–2429 (1968). https://doi.org/10.2514/3.5008
7. Fragkos, K.B., Papoutsis-Kiachagias, E.M., Giannakoglou, K.C.: pFOSM: An efficient algorithm for aerodynamic robust design based on continuous adjoint and matrix-vector products. Comput. Fluids **181**, 57–66 (2019)
8. Haldar, A., Mahadevan, S.: Probability, Reliability and Statistical Methods in Engineering Design. John Wiley & Sons (1999)
9. Kriegesmann, B., Lüdeker, J.K.: Robust compliance topology optimization using the first-order second-moment method. Struct. Multidiscip. Optim. **60**(1), 269–286 (2019). https://doi.org/10.1007/s00158-019-02216-8
10. Lazarov, B.S., Schevenels, M., Sigmund, O.: Topology optimization with geometric uncertainties by perturbation techniques. Int. J. Numer. Meth. Eng. **90**(11), 1321–1336 (2012). https://doi.org/10.1002/nme.3361
11. Nelson, R.B.: Simplified calculation of eigenvector derivatives. AIAA J. **14**(9), 1201–1205 (1976). https://doi.org/10.2514/3.7211
12. Rosen, J.B.: The gradient projection method for nonlinear programming, Part II, nonlinear constraints. J. Soc. Industrial Appli. Math. **9**(4), 514–532 (1961). https://doi.org/10.1137/0109044
13. Schevenels, M., Lazarov, B.S., Sigmund, O.: Robust topology optimization accounting for spatially varying manufacturing errors. Comput. Methods Appl. Mech. Eng. **200**(49–52), 3613–3627 (2011). https://doi.org/10.1016/j.cma.2011.08.006
14. Seyranian, A.P., Lund, E., Olhoff, N.: Multiple eigenvalues in structural optimization problems. Structural optimization **8**(4), 207–227 (1994). https://doi.org/10.1007/BF01742705
15. Sigmund, O.: Manufacturing tolerant topology optimization. Acta. Mech. Sin. **25**(2), 227–239 (2009). https://doi.org/10.1007/s10409-009-0240-z
16. Sigmund, O.: On the usefulness of non-gradient approaches in topology optimization. Struct. Multidiscip. Optim. **43**(5), 589–596 (2011). https://doi.org/10.1007/s00158-011-0638-7
17. Steltner, K., Pedersen, C.B.W., Kriegesmann, B.: Semi-intrusive approach for stiffness and strength topology optimization under uncertainty. Optim. Eng. (2022). https://doi.org/10.1007/s11081-022-09770-z
18. Wang, F., Lazarov, B.S., Sigmund, O.: On projection methods, convergence and robust formulations in topology optimization. Struct. Multidiscip. Optim. **43**(6), 767–784 (2011). https://doi.org/10.1007/s00158-010-0602-y
19. Yao, W., Chen, X., Luo, W., van Tooren, M., Guo, J.: Review of uncertainty-based multidisciplinary design optimization methods for aerospace vehicles. Prog. Aerosp. Sci. **47**(6), 450–479 (2011). https://doi.org/10.1016/j.paerosci.2011.05.001

Optimal Control

Dynamic Analysis on Formation for the Flight Energy-Saving of a Flapping Wing Robots Flock

Yichao Shen[1,2], Yuanzhe Cui[2], Wei Zhu[2], Peter Eberhard[1(✉)], and Qirong Tang[2,3(✉)]

[1] Institute of Engineering and Computational Mechanics, University of Stuttgart, Stuttgart, Germany
peter.eberhard@itm.uni-stuttgart.de
[2] Laboratory of Robotics and Multibody System, School of Mechanical Engineering, Tongji University, Shanghai, People's Republic of China
qirong.tang@outlook.com
[3] Chinesisch Deutsches Hochschulkolleg, Tongji University, Shanghai, People's Republic of China

Abstract. The flight of a flapping wing robot has advantages of low noise and flexibility. A flapping wing robot in a formation can achieve more lift through the upwash effect generated by other robots to realize energy saving. In this paper, the wing tip vortex model is established and the flight formation of the flapping wing robots is analyzed in the simulation. The flapping frequency, transverse distance, vertical distance, and the phase difference are simulated to study the influence on the lift and drag of a trailing robot, which helps to estimate the consumption of the energy. According to the simulation, the overall energy consumption can be reduced under the flight formation which takes aerodynamic forces into account.

1 Introduction

In recent years, flapping wing robot systems became an interesting direction in the robotics field. Flapping wing robots have advantages of high aerodynamic efficiency, low flight noise, high maneuverability, and robustness to disturbances. Researchers have done a lot of work on the stability of the control system and the reduction of the energy consumption. In [1], an approach is proposed to analyze the dynamic stability and develop trajectory-tracking controllers for flapping-wing micro robots. They used a multibody dynamics simulation framework to accomplish various flight objectives and energy-saving. In [2], control actions are designed aiming at increasing the flight performances. In [3], the formation of some groups of birds are investigated and results are presented that aerodynamic performance of the formation flight would improve when introducing active morphing. In [4], mechanism design and lift force calculation of an underactuated

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 87–98, 2024.
https://doi.org/10.1007/978-3-031-50000-8_8

flapping wing robot is investigated and an approach for quantitatively calculating lift and thrust forces of the underactuated flapping wing system is proposed. In [5], the advantages of wing twist and fold for flapping wing robots is explored and a new dynamic model is developed which could reduce the overall power consumption. In [6], a multiple modes flight control method is proposed and a fuzzy control strategy is presented to realize flight performance with higher accuracy and higher flexibility.

Nowadays, it is hard to disentangle the complexity of the emerging fluid dynamics by current analytical understanding alone. In order to achieve energy-saving flight control, an optimized flight formation and control framework is proposed to drive a flapping wing robot flock.

2 Proposed Method

The model of the flapping wing robot is show in Fig. 1. The wingspan is 1.6 m, the body length of the robot is 0.931 m.

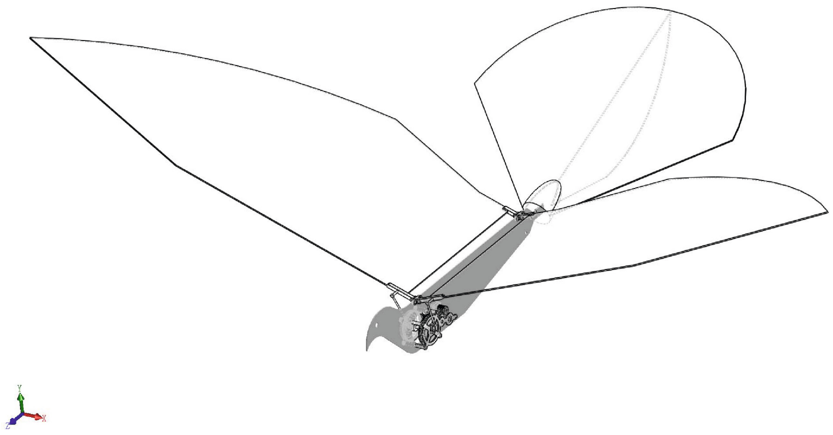
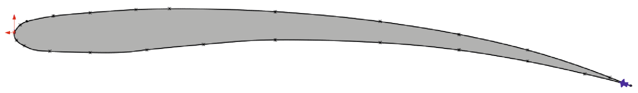


Fig. 1. The model of the flapping wing robot.

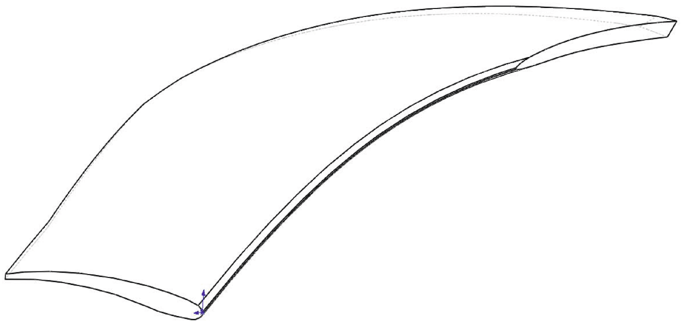
The shape of the robot wing has a great influence on the lift. According to the birds in nature, the wing of birds is usually thick in the leading edge and thin in the trailing edge. Also the upward curved and arched shape of the leading edge can effectively provide lift. For flapping wing robots with a wingspan greater than 0.3 m, the use of a bird wing can effectively increase lift, so the surface is usually designed as a curved surface. Therefore, the shape of the wing is improved as shown in Fig. 2.

The two robots formation flight model is shown in Fig. 3. According to the wing-tip vortex model, the velocity induced by the wing-tip vortex is given by

$$\mathbf{U} = \frac{\boldsymbol{\Phi}\varGamma}{4\pi|\mathbf{r_c}|}\left(\cos\beta_1 - \cos\beta_2\right), \tag{1}$$



(a) The sectional view of the wing.



(b) The model of the left wing.

Fig. 2. Model of the improved wing.

where $\mathbf{r_c}$ is a distance from the vortex line to a point O on the wing, Γ is the strength of bound vortex, $\mathbf{\Phi}$ is the unit vector orthogonal to $\mathbf{r_c}$. When the longitudinal distance between the leading robot and the trailing robot is more than twice longer than the wingspan, it can be considered that $\beta_1 \approx 0$, $\beta_2 \approx \pi$, so Eq. (1) can be simplified as

$$\mathbf{U} = \frac{\mathbf{\Phi}\Gamma}{2\pi r_c}. \tag{2}$$

The strength of bound vortex is given according to the Kutta-Joukowski law [7].

$$\Gamma = \frac{L}{\rho U b'} = \frac{1/2\rho U^2 S C_L}{\rho U\left(\pi/4\right) b} = \frac{2S}{\pi b^2} U C_L b = \frac{2}{\pi A_R} U C_L b, \tag{3}$$

where U is the fluid velocity, b is the wing span, ρ is the fluid density, C_L is the lift coefficient, S is the wing area, $A_R = b^2/S$ is the aspect ratio, $b' = \pi b/4$, $\bar{y}$ is the transverse distance, and $\bar{z}$ is used to express the altitude difference between two robots. So the velocity induced by the y-direction upwash flow on the trailing robot is given by

$$U_y = \frac{\Gamma}{2\pi}\left[\frac{\left(\bar{y} - \frac{b'}{2} - y\right)^2}{\sqrt{\left(\bar{y} - \frac{b'}{2} - y\right) + \bar{z}^2}} - \frac{\left(\bar{y} + \frac{b'}{2} - y\right)^2}{\sqrt{\left(\bar{y} + \frac{b'}{2} - y\right) + \bar{z}^2}}\right]. \tag{4}$$

The average velocity induced by the y-direction upwash flow can be calculated as

$$\overline{U}_y = \frac{\Gamma}{4\pi b}\left[\ln \frac{y'^2+z'^2}{\left(y'-\frac{\pi}{4}\right)^2 - z'^2} - \ln \frac{\left(y'+\frac{\pi}{4}\right)^2+z'^2}{y'^2+z'^2}\right], \tag{5}$$

where $y' = \bar{y}/b$, $z' = \bar{z}/b$.

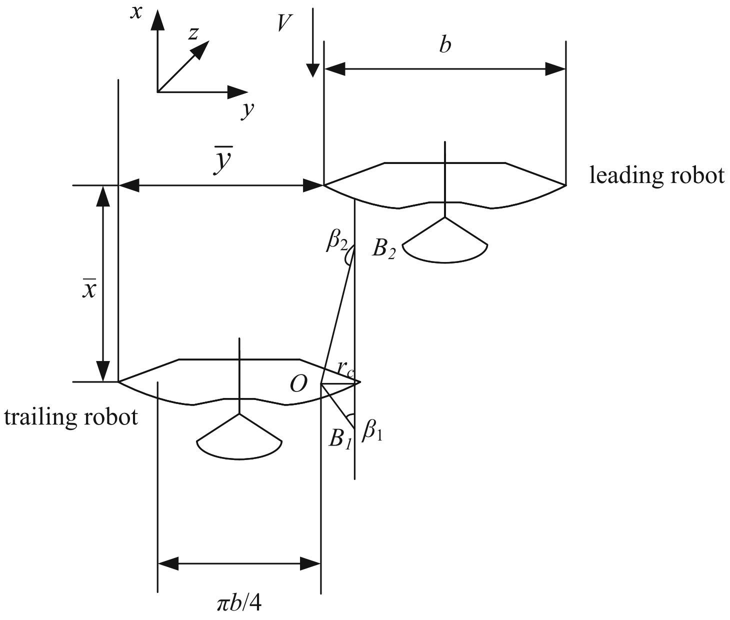


Fig. 3. Top view of vortex model for two robot formation flight.

In order to further analyze the influence of the trailing robot affected by the wingtip vortex of the leading robot, Fig. 4 presents a side view of the trailing robot wing.

The total lift L_F and the total drag D_F of the trailing robot lead to

$$L_F = L'\cos(\Delta\zeta) + \Delta L = L'\cos(\Delta\zeta) + D'\sin(\Delta\zeta), \tag{6}$$

$$D_F = D'\cos(\Delta\zeta) - \Delta D = D'\cos(\Delta\zeta) - L'\sin(\Delta\zeta), \tag{7}$$

where L' and D' are the additional lift and drag produced by the upwash flow, $\Delta\zeta$ is the change of the attack angle influenced by the upwash flow, ΔD is the change of the drag, and ΔL is the change of the lift. Because lift is far bigger than drag when flying, $\Delta\zeta$ is a very small angle, and one can get

$$\Delta\zeta = \arctan\left(\frac{|\bar{U}_y|}{U}\right) \approx \frac{|\bar{U}_y|}{U}. \tag{8}$$

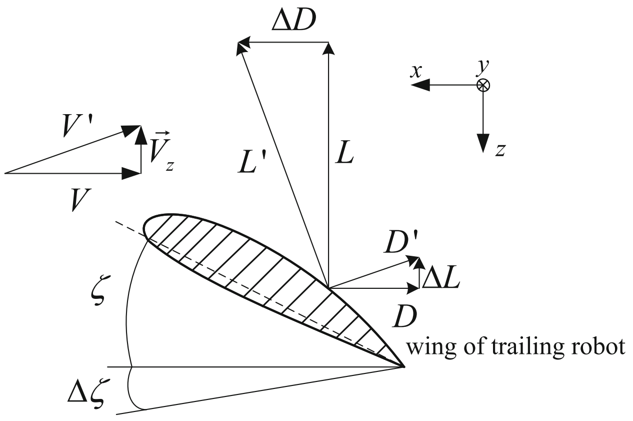


Fig. 4. The side view of the trailing robot wing.

In this time, the lift increment equals to

$$\Delta L = qS\Delta C_L, \tag{9}$$

where q is the dynamic pressure, ΔC_L represents the lift increment coefficient, and the lift increment coefficient is

$$\begin{aligned} \Delta C_L &= \Delta\zeta a_W \\ &\approx \frac{2C_L a_W}{\pi A_R \pi^2}\left[\ln\frac{y'^2+z'^2+\mu^2}{\left(y'-\frac{\pi}{4}\right)^2+z'^2+\mu^2} - \ln\frac{\left(y'+\frac{\pi}{4}\right)^2+z'^2+\mu^2}{y'^2+z'^2+\mu^2}\right], \end{aligned} \tag{10}$$

where a_W is the change rate of the lift curve, μ is a constant related to vortex.

The drag increment equals to

$$\Delta D \approx L_F\left(\frac{|\bar{U}_y|}{U}\right). \tag{11}$$

Finally, the drag increment coefficient can be found, which is

$$\Delta C_D = \frac{\Delta D}{qS} \approx \frac{2C_L}{\pi A_R \pi^2}\left[\ln\frac{y'^2+z'^2+\mu^2}{\left(y'-\frac{\pi}{4}\right)^2+z'^2+\mu^2} - \ln\frac{\left(y'+\frac{\pi}{4}\right)^2+z'^2+\mu^2}{y'^2+z'^2+\mu^2}\right]. \tag{12}$$

According to Eqs. (10) and (12), the relationship between lift, drag, transverse distance y, and altitude difference z can be completed. After calculating, when $y' = \pi/4$, namely $y = \pi b/4$, and $z' = 0$, namely altitude difference $z = 0$, ΔL is biggest and ΔD is smallest. Considering Eqs. (6) and (7), the lift L_F is the biggest and the drag D_F is the smallest, so it is in the best formation position for the trailing robot according to these considerations. The formation shape will be tested in the simulation in the next section.

The proposed framework consists of a PI (Proportional-Integral) velocity controller and a formation controller, which takes into account the aerodynamic force generated by multiple flapping and twisting motions under the condition of formation flight. The formation controller utilizes the Markov Decision Process model [8] to provide formation policy. A flapping wing robot flock achieves energy-saving flight using the aerodynamic analysis of rolling moment, L_{wake}, and average upwash, $\overline{W}_{wake}$. Generally, the inflow velocity is the reference variable. A block diagram of the framework is illustrated in Fig. 5.

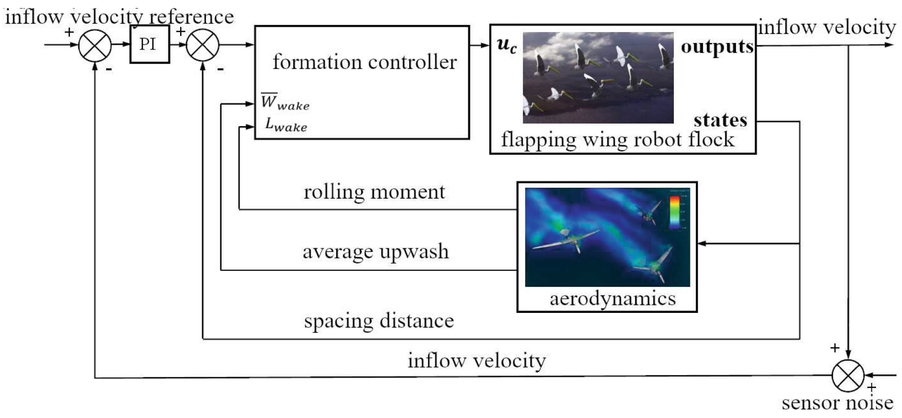


Fig. 5. Block diagram of the proposed optimized control framework.

3 Simulations

3.1 A Single Flapping Wing Robot

In this section, the aerodynamic simulation of a single flapping wing robot is done to obtain the lift and drag under different flapping wing frequencies, so the effective flapping frequency for energy saving can be found.

The flapping frequency of the flapping wing robot is directly related to the lift and drag, and has a certain positive correlation. This section will study and simulate the relationship between the wing flapping frequency and lift-drag ratio. Therefore, the whole simulation process should focus on the change of lift and drag on the wing of flapping wing robot.

As shown in Fig. 6, to make the simflow to reality, a cuboid wind tunnel is built in simulation and it is 4 m long, 3 m wide and 2.4 m high. The incoming flow velocity at the entrance of the wind tunnel is set to 6 m/s in z-axis direction which represents the robot flight speed. The flapping wing robot is in the center of the wind tunnel.

Considering the flapping frequency of pelican birds in nature, the range is about 2–5 Hz. The wingspan of the flapping wing robot b is 1.6 m and the inflow

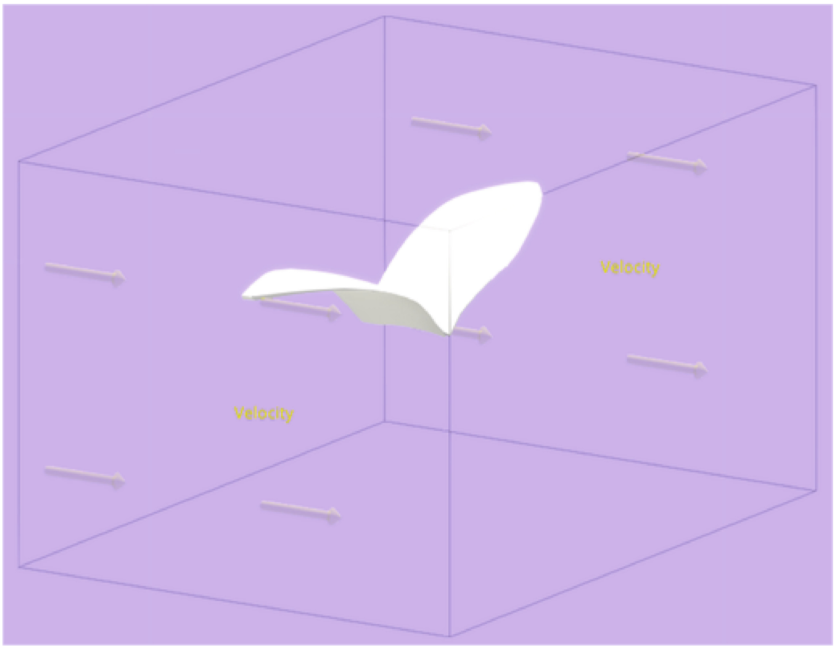


Fig. 6. The simulation environment.

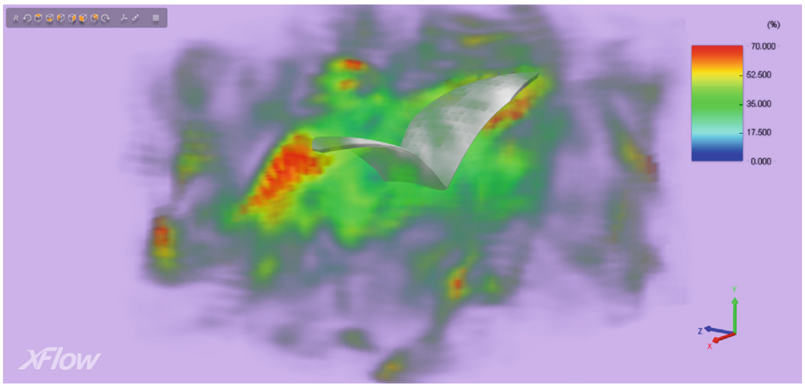


Fig. 7. The vortex intensity around the single flapping wing robot.

velocity is 6 m/s in the wind tunnel. The simulations with different flapping frequencies are conducted in XFlow, which is shown in Fig. 7, and the simulation results of lift and drag are shown in Fig. 8.

According to Fig. 8, the lift and drag are influenced by the flapping frequency. The average lift and drag of a single flapping wing robot at 1.5 Hz, 3 Hz, and 4.5 Hz are obtained, as shown in Table 1. Since the lift and drag increase with the increase of flapping frequency, the concept of lift drag-ratio is introduced to get an idea about the energy consumption of a flapping wing robot.

Table 1. Relationship between flapping frequency and lift-drag ratio.

	1.5 Hz	3 Hz	4.5 Hz
lift (N)	0.9627	1.4726	1.9787
drag (N)	0.8353	1.1891	1.4667
lift-drag ratio	1.1525	1.2385	1.3491

Table 1 shows that when the flapping frequency is 4.5 Hz, the lift-drag ratio of flapping wing robot is the largest, and the effect of energy saving is the most obvious. Also, considering the large birds in nature like pelicans, their flapping frequency is often smaller than 5 Hz. Therefore, the formation flight simulation in the following section will be carried out for this flapping frequency 4.5 Hz.

3.2 Double Flapping Wing Robots

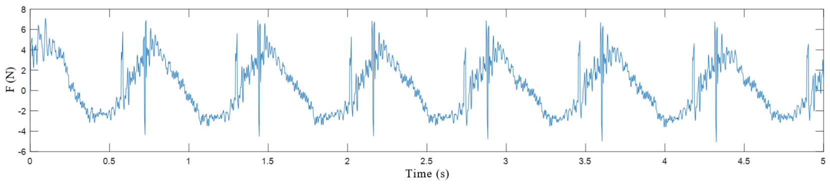
In formation flight simulation, the flapping frequency is set to 4.5 Hz to explore a energy-saving formation flight formation. The additional lift and drag caused by the vortex effect of the leading robot on the trailing robot are influenced by the transverse distance and the vertical distance.

In order to find which transverse distance is most influential on the energy saving, the wind tunnel is the same as the one in the single flapping wing robot simulation, and the inflow velocity is 6 m/s. The two robots are in the wind tunnel with different transverse distances. The lift and drag of the leading robot and the trailing robot are measured and the calculated average is shown in Table 2. The simulation time is set to 2 s.

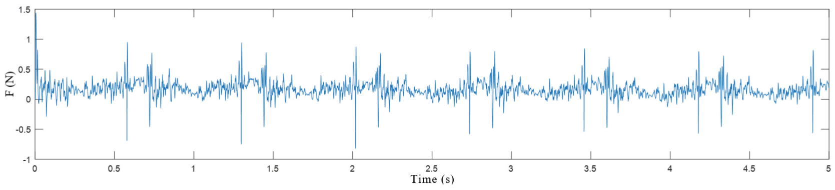
Table 2. The effect of the different transverse distances on the lift and drag.

	0	$\pi b/8$	$\pi b/4$	$3\pi b/8$
lift_{leader} (N)	0.7955	0.7230	1.6760	1.7411
$\text{lift}_{follower}$ (N)	1.1861	1.8096	1.7732	1.7884
drag_{leader} (N)	0.6780	0.6299	0.6694	0.7020
$\text{drag}_{follower}$ (N)	0.5997	0.5993	0.6270	0.6222
lift-drag ratio_{leader}	1.1733	1.1478	2.5038	2.4803
lift-drag $\text{ratio}_{follower}$	1.9779	3.0195	2.8279	2.8745

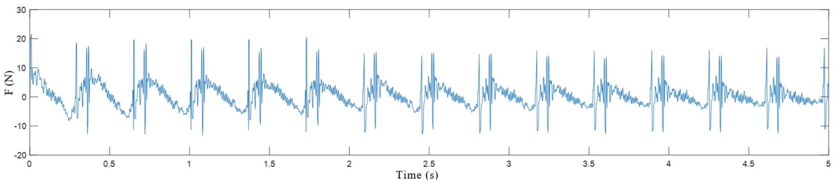
Table 2 shows that the lift of the trailing robot is bigger than the one of the leading robot while the drag of the trailing robot is smaller than the one



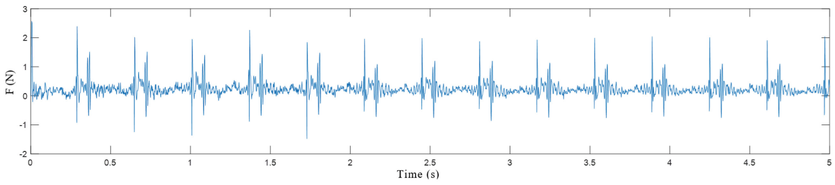
(a) Lift at 1.5 Hz flapping frequency.



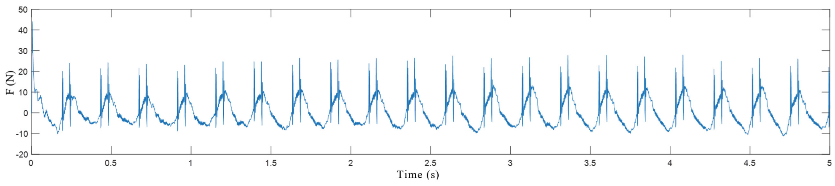
(b) Drag at 1.5 Hz flapping frequency.



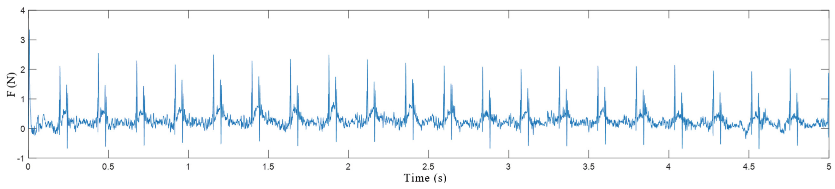
(c) Lift at 3 Hz flapping frequency.



(d) Drag at 3 Hz flapping frequency.



(e) Lift at 4.5 Hz flapping frequency.



(f) Drag at 4.5 Hz flapping frequency.

Fig. 8. Lift and drag of a flapping wing robot under different flapping frequencies.

of the leading robot in the simulation, which means that the trailing robot can obtain additional lift and reduce the drag by the vortex of the leading robot. It can also be seen from the table that the lift-drag ratio of the trailing robot increases quickly from 0 to $\pi b/8$ and the lift-drag ratio of the leading robot increases quickly from 0 to $\pi b/4$. Considering the consumption of the energy and the formation flight, choose $\pi b/4$ as the transverse distance between the leading robot and the trailing robot.

In order to study the influence of the phase difference between leading robot and the trailing robot, the phase difference is set at 0, $\pi/2$, π, $3\pi/2$ and the transverse distance is set at $\pi b/4$. The lift and drag of the leading robot and the trailing robot are measured and the calculated average is shown in Table 3.

Table 3. The effect of the different phase difference on the lift and drag.

	0	$\pi/2$	π	$3\pi/2$
lift$_{leader}$ (N)	1.6760	1.4062	2.0084	1.7611
lift$_{follower}$ (N)	1.7732	2.0786	0.7926	1.4939
drag$_{leader}$ (N)	0.6694	0.6571	0.6983	0.6805
drag$_{follower}$ (N)	0.6270	0.6111	0.6074	0.6485
lift-drag ratio$_{leader}$	2.5038	2.1400	3.3065	2.5878
lift-drag ratio$_{follower}$	2.8279	3.4014	1.3049	2.3036

Table 3 shows that when the phase difference is $\pi/2$, the lift of the leading robot is much smaller than the one of the trailing robot and the drag of the leading robot is bigger than the one of the trailing robot. The lift-drag ratio increases much when the phase difference achieves $\pi/2$. So when the transverse distance is $\pi b/4$ and phase difference is $\pi/2$, the lift-drag ratio of the trailing robot is much bigger than others, which means here that the consumption of the energy is much smaller.

Finally, the transverse distance between the leading robot and the trailing robot is set at $\pi b/4$, and the phase difference is set at $\pi/2$. The vertical distance between the leading robot and the trailing robot is set at 0 m, 0.15 m and 0.3 m to study the influence of the vertical distance between two robots on energy saving. Also the wind tunnel is using the same inflow velocity at 6 m/s. The two robots are in the wind tunnel with different vertical distances. The lift and drag of the leading robot and the trailing robot are measured and the calculated average is shown in Table 4.

According to Table 4, the lift of the trailing robot decreases with the increase of the vertical distance between leading robot and the trailing robot. Also, the drag increases and the lift-drag ratio decreases. So when the vertical distance is 0 m, the trailing robot has the biggest lift and the smallest drag which means the consumption of energy is the smallest.

Table 4. The effect of the different vertical distance on the lift and drag.

	0 m	0.15 m	0.3 m
lift (N)	1.7730	1.2392	1.0630
drag (N)	0.6270	0.6760	0.6952
lift-drag ratio	2.8279	1.2392	1.0630

4 Discussion

In the present study, flapping wing robots are simulated to study the consumption of energy. We can find that the distances, frequency or phase difference can have an impact on energy saving. In two flapping wing robots, the leading robot can have the lift with 1.4062 N and the trailing robot will have the lift with 2.0786 N, so the average lift is 1.7424 N, which means each flapping wing robot can increase the lift by 23% comparing to a single flapping wing robot, and it can also save up to 23% of energy consumption. In further research, the flight formation will contain more flapping wing robots as proposed in Fig. 9, and a further study of the flight formation will analyzed which takes aerodynamic forces into account.

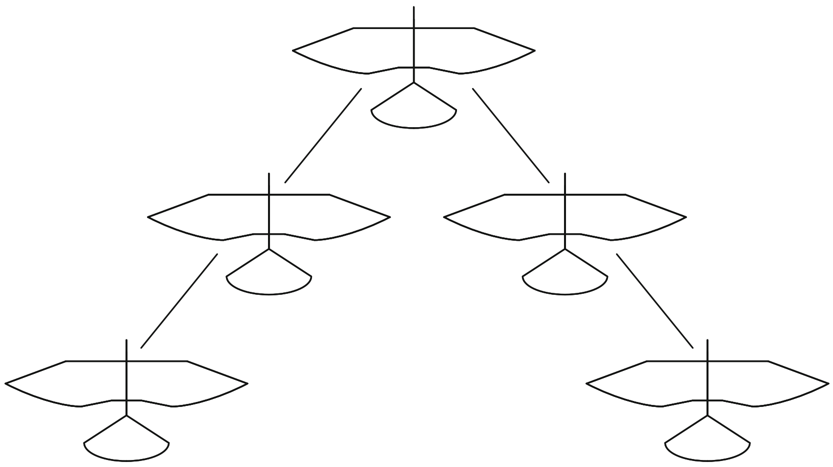


Fig. 9. Formation of the flapping wing robots.

5 Conclusions

In nature, the formation flight of a pelican flock can effectively reduce the energy consumption of individuals. In this study, the transverse distance, vertical distance, phase difference and flapping frequency are analyzed with two flapping wing robots for measuring the lift and drag. Through the simulation, it is feasible to reduce the energy consumption of swarm flight of flapping wing robots by a formation controller. This idea provides a feasible solution for the formation control of flapping wing robots with optimal energy consumption in formation flight, and also provides an expanding direction for research in this field.

References

1. Bhatti, M.Y., Lee, S.G., Han, J.H.: Dynamic stability and flight control of biomimetic flapping-wing micro air vehicle. Aerospace **8**(12), 362 (2021)
2. Califano, F., et al.: Decoding and realising flapping flight with Port-Hamiltonian system theory. Annu. Rev. Control. **51**, 37–46 (2021)
3. Billingsley, E., Ghommem, M., Vasconcellos, R., Abdelkefi, A.: Optimal aerodynamic design of multi-flapping wing vehicles with morphing capabilities. In: AIAA Scitech 2021 Forum (2021)
4. Sun, W., Yu, J., He, G., Cai, Y.: Study on transmission mechanism and flexible flapping wings of an underactuated flapping wing robot. J. Intell. Robot. Syst. **104**(2), 1–13 (2022)
5. Fan, X., Breuer, K., Vejdani, H.: Wing fold and twist greatly improves flight efficiency for bat-scale flapping wing robots. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems, Prague, Czech Republic, pp. 7391–7397 (2021)
6. Xu, W., Pan, E., Liu, J., Li, Y., Yuan, H.: Flight control of a large-scale flapping-wing flying robotic bird: system development and flight experiment. J. Intell. Robot. Syst. **35**(2), 235–249 (2022)
7. Proud, A., Pachter, M., D'Azzo, J.: Close formation flight control. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, p. 4207 (1999)
8. Alroobaea, R., Binmahfoudh, A., Alzahrani, S.M., Althobaiti, A.: Markov decision process with deep reinforcement learning for robotics data offloading in cloud network. J. Electron. Imaging **31**(6), 061809 (2022)

Optimal Control of Open–Loop Multibody Systems Recovered from Data

Maciej Pikuliński(✉) and Paweł Malczyk

Faculty of Power and Aeronautical Engineering, Institute of Aeronautics and Applied Mechanics, Warsaw University of Technology, Nowowiejska 24, 00-665 Warsaw, Poland
{maciej.pikulinski.dokt,pawel.malczyk}@pw.edu.pl

Abstract. In this paper, we consider data–driven models of open–loop multi–rigid–body systems. Such models may possess uncertain coefficients or additional state–dependent terms, which are not present in the nominal system. This may cause performance issues when the adjoint–based optimal control is applied. A sample test case illustrates the robustness of the optimal control with respect to modeling errors. This paper's paramount goal is to discuss results obtained from applying optimal control adjoint-based scheme to an open–loop multi–rigid–body system, the model of which was identified from data.

1 Introduction

Identification of unknown multibody system (MBS) dynamics from given experimental or numerical data is of paramount importance for accurate future state predictions or the application of advanced model-based control strategies. A traditional way to identify an MBS's dynamics is to find its linear model parameters [6]. On the other hand, new approaches emerge from the recent development of identification with the use of techniques inspired by machine learning [1,7]. One of these techniques is the least-squares regression with L^1 regularization, which determines parameters and, more importantly, the sparse structure of the model. Sparsity means that the identified model tends to consist of a minimal number of governing equations. Such approaches have not yet been widely used in multibody dynamics.

Solution of optimal control problem for MBS might be approached using gradient-based optimization methods. In order to apply these techniques, one needs to know the gradient of the cost function, whose analytical derivation renders difficulties in the field of MBS. Moreover, numerical approaches for finding the gradient pose a simple task neither and are constantly developed. Among these methods, the finite difference method and the direct differentiation method are often proposed. Nonetheless, in the case of multivariate optimization (a significant number of decisive variables might be called in control of MBS) and large-scale models, the two mentioned methods tend to perform worse than the adjoint method [3], which recently raised scientists' interest in its use with different approaches to dynamics description [10,11].

Many advancements have been done in the field of data–driven identification. Sparse regression is used to discover equations of motion and create parsimonious models with the fewest terms out of the candidates. Unfortunately, the adjoint–based optimal control may not be robust with respect to induced modeling errors.

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 99–109, 2024.
https://doi.org/10.1007/978-3-031-50000-8_9

The primary importance of this paper is to discuss the results of optimal control of fully actuated, open–loop multi–rigid–body systems, when a model is identified from data. We compare the results of optimal control of open-chain structure MBS in two scenarios, which differ in the model of MBS used in the optimization.

A nominal model is assumed to be perfectly aligned with a hypothetic real MBS. In the first scenario, the nominal model is known and used to design control signals to control the MBS. The other scenario must first acquire the model by identifying it from data recorded during the nominal model's simulations. Then, the identified model is used in the optimization process, and eventually, the control signals are applied to simulation with the nominal model. In the end, it is discussed how the inaccuracy of the model obtained from the identification affects overall control results.

The paper is outlined as follows. In Sect. 2, we briefly recall MBS dynamics described with joint coordinates. Then, we introduce the concept of identification based on regularized regression in Sect. 3. Further, the application of the adjoint method for optimal control of MBS follows in Sect. 4, and, in the end, we describe compared scenarios and discuss the results in Sect. 5. Section 6 concisely sums up the presented work and outlines further research directions.

2 Multibody Dynamics in Joint Coordinates

Many methods exist to model a constrained system of bodies, describe the bodies' spatial relations, and compute their coordinates in the global inertial frame. Each method might be better or worse for particular tasks and has different pros and cons depending on the application. From all these approaches, we focus on the automatic generation of equations of motion (EOM) using joint coordinates (independent coordinates), which lead to a minimal set of coordinates. In the case of open-chain MBS, the number of coordinates is equal to the degrees of freedom of the MBS. Furthermore, the literature reveals sequential as well as parallel algorithms for recursive and, at the same time, efficient generation of EOMs for such an approach [4,5,9].

In this work, without loss of generality, we consider planar and open kinematic chains consisting of n elements and the same number of degrees of freedom (n). We define the vector of absolute coordinates

$$\mathbf{p} = \left[\mathbf{p}_1^T \ \mathbf{p}_2^T \ \cdots \ \mathbf{p}_n^T\right]^T \in \mathbb{R}^{3n}, \tag{1}$$

which describes the center of mass of individual bodies and the rotation of the local system associated with each body relative to the global frame, e.g., the following absolute coordinates vector describes the i-th body $\mathbf{p}_i = \left[x_i \ y_i \ \varphi_i\right]^T$. Additionally, we define a corresponding vector of velocities as follows

$$\mathbf{V} = \dot{\mathbf{p}} = \left[\mathbf{V}_1^T \ \mathbf{V}_2^T \ \ldots \ \mathbf{V}_n^T\right]^T \in \mathbb{R}^{3n}. \tag{2}$$

Constraints, the kinematic pairs in our understanding, are formulated by m independent constraints equations and aggregated as

$$\boldsymbol{\Phi}(\mathbf{p}) = \mathbf{0} \in \mathbb{R}^m. \tag{3}$$

Then, having the environment defined, we can express Newton-Euler equations of motions for MBS with an open kinematic chain structure and absolute coordinates as follows

$$\begin{cases} \mathbf{M}\dot{\mathbf{V}} + \mathbf{D}^T\boldsymbol{\lambda} = \mathbf{Q} + \mathbf{Q}^{(\mathbf{u})} \\ \boldsymbol{\Phi}(\mathbf{p}) = \mathbf{0} \quad \rightarrow \quad \ddot{\boldsymbol{\Phi}} \equiv \mathbf{D}\dot{\mathbf{V}} + \dot{\mathbf{D}}\mathbf{V} = \mathbf{0}, \end{cases} \tag{4}$$

where $\mathbf{M}(\mathbf{p}) \in \mathbb{R}^{3n\times 3n}$ is a mass matrix of the whole system, $\mathbf{V} \in \mathbb{R}^{3n}$ is the previously defined vector of linear and angular velocity, $\mathbf{D}(\mathbf{p}) \in \mathbb{R}^{m\times 3n}$ is the Jacobi matrix indicating unavailable directions of motions, $\boldsymbol{\lambda} \in \mathbb{R}^m$ is the Lagrange coefficients vector, $\mathbf{Q}(\mathbf{p}, \dot{\mathbf{p}}) \in \mathbb{R}^{3n}$ are external forces and torques vector, and $\mathbf{Q}^{(\mathbf{u})} \in \mathbb{R}^{3n}$ represents driving forces and torques. It is worth pointing out that (4) poses a differential-algebraic equation (DAE), which in general needs to be solved with special techniques tailored for the structure of the equation.

Now, let us transform the first equation to be compatible with joint coordinates $\mathbf{q} \in \mathbb{R}^n$, which, e.g., could be easily defined as angles of rotations for each kinematic pair (assumed we have only revolute joints). There is a linear mapping between $\dot{\mathbf{q}}$ and $\mathbf{V}$ [5].

$$\mathbf{V} = \mathbf{H}\dot{\mathbf{q}} \quad \rightarrow \quad \dot{\mathbf{V}} = \mathbf{H}\ddot{\mathbf{q}} + \dot{\mathbf{H}}\dot{\mathbf{q}}, \tag{5}$$

where $\mathbf{H}(\mathbf{q}) \in \mathbb{R}^{3n\times n}$ is the joint motion subspace dependent (typically) on the MBS kinematic configuration. An important fact is that the motion subspace $\mathbf{H}$ is orthogonal to the Jacobi matrix $\mathbf{D}$, i.e., $\mathbf{D}\cdot\mathbf{H} = \mathbf{0}$. By inserting (5) into (4) and multiplying the result by $\mathbf{H}^T$, we obtain the equation of motion formulated for joint coordinates

$$\left(\mathbf{H}^T\mathbf{M}\mathbf{H}\right)\ddot{\mathbf{q}} - \mathbf{H}^T\left(\mathbf{Q} - \mathbf{M}\dot{\mathbf{H}}\dot{\mathbf{q}}\right) = \mathbf{u} \quad \rightarrow \quad \mathscr{M}(\mathbf{q})\ddot{\mathbf{q}} - \mathscr{F}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{u}, \tag{6}$$

where $\mathscr{M}(\mathbf{q}) = \mathbf{H}^T\mathbf{M}\mathbf{H}$ is the mass matrix, $\mathscr{F}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{H}^T\left(\mathbf{Q} - \mathbf{M}\dot{\mathbf{H}}\dot{\mathbf{q}}\right)$ is the vector of forces and torques, and $\mathbf{u} = \mathbf{H}^T\mathbf{Q}^{(\mathbf{u})}$ is the vector of joint driving forces and torques. Compared to the absolute coordinates, the benefit of the joint coordinates approach is easily seen as (6) is an ordinary differential equation, whereas (4) is a DAE.

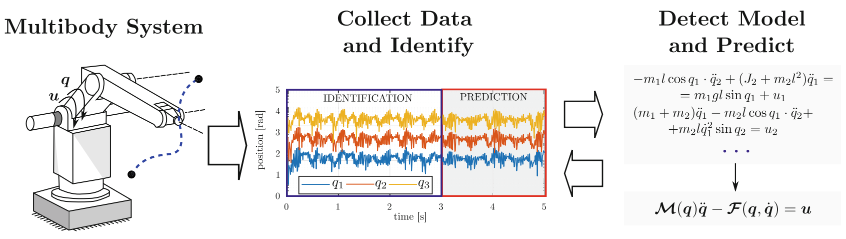

Multibody System
q
u
Collect Data and Identify
IDENTIFICATION
PREDICTION
position [rad]
time [s]
q_1 q_2 q_3
Detect Model and Predict
$-m_1 l\cos q_1 \cdot \ddot{q}_2 + (J_2 + m_2 l^2)\ddot{q}_1 = = m_1 g l \sin q_1 + u_1$
$(m_1 + m_2)\ddot{q}_1 - m_2 l\cos q_1 \cdot \ddot{q}_2 + +m_2 l\dot{q}_1^2 \sin q_2 = u_2$
$\mathcal{M}(q)\ddot{q} - \mathcal{F}(q,\dot{q}) = u$

Fig. 1. Identification workflow

3 Identification Based on Regularized Regression

In this work, we implement the least-squares regression with regularization to identify an MBS's structure and parameters from previous simulation data (generally – recorded measurements). In Fig. 1 a scheme of the identification workflow can be seen. This scheme corresponds to further subsections that describe consecutive steps in the process.

3.1 Collect Data (Simulation, Measurements)

The process begins with gathering measurements, including control signals $\mathbf{u}_i = [u_i(t_1) \dots u_i(t_k)]^T$ and joint rotations, which, in our case, are equal to the set of joint coordinates $\mathbf{q}_i = [q_i(t_1) \dots q_i(t_k)]^T$, $i = 1, \dots, n$. The data is collected with a constant sampling time from t_1 to t_k such that $k \gg n$. If raw velocity data is unavailable, it, along with accelerations and higher-order derivatives, can be found with numerical differentiation techniques. Ultimately, all the measurements can be organized as the following matrices

$$\mathbf{U} = \begin{bmatrix} | & & | \\ \mathbf{u}_1 & \cdots & \mathbf{u}_n \\ | & & | \end{bmatrix} \in \mathbb{R}^{k \times n}, \quad \mathbf{Y} = \begin{bmatrix} | & & | & | & & | & | & & | \\ \mathbf{q}_1 & \cdots & \mathbf{q}_n & \dot{\mathbf{q}}_1 & \cdots & \dot{\mathbf{q}}_n & \ddot{\mathbf{q}}_1 & \cdots & \ddot{\mathbf{q}}_n \\ | & & | & | & & | & | & & | \end{bmatrix} \in \mathbb{R}^{k \times 3n}. \tag{7}$$

3.2 General Dynamics Model of MBS

It is assumed that in each time step, the dynamics of MBS can be approximated by a linear combination of p columns from a library of base functions $\theta = \theta(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ ($1 \times p$ vector) and an unknown coefficient vector $\boldsymbol{\xi}_i \in \mathbb{R}^p$, $i = 1, \dots, n$ (a column vector) determining a share of each base function in resulting equations of motions

$$\mathbf{u} = \mathscr{M}(\mathbf{q})\ddot{\mathbf{q}} - \mathscr{F}(\mathbf{q}, \dot{\mathbf{q}}) \approx (\theta \cdot \boldsymbol{\xi})^T \quad \rightarrow \quad (\theta \cdot \boldsymbol{\xi})^T - \mathbf{u} = \mathbf{0}, \tag{8}$$

where $\boldsymbol{\xi} = [\boldsymbol{\xi}_1 \dots \boldsymbol{\xi}_n] \in \mathbb{R}^{p \times n}$ is a matrix of stacked unknown coefficient vectors $\boldsymbol{\xi}_i$ which are subject to optimization (regression with regularization). Proper formulation and choice of the base functions library $\theta = \theta(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ is a cumbersome task as there are multiple approaches to fill it with functions, e.g., harmonic functions from Fourier series or polynomial functions and their non-linear combinations. Nonetheless, it is worth pointing out that familiarity with motion equations of MBS might help with the selection.

3.3 Computing $\boldsymbol{\xi}$ Coefficient with Regularized Regression

Let us think of (8) as a dynamics model of MBS valid for just a single time step. We can expand this idea with the previously collected measurement data into a set of dynamics models interrelated to individual steps from t_1 to t_k. In other words, we generate linear dynamics models (8) with respect to $\boldsymbol{\xi}$ for each recorded time step t_i. The generated batch of models can be written succinctly in the following linear form

$$\mathbf{U} = \boldsymbol{\Theta}(\mathbf{Y})\boldsymbol{\xi}. \tag{9}$$

Matrix $\boldsymbol{\Theta}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) = \left[\theta_1^T \dots \theta_k^T\right]^T \in \mathbb{R}^{k \times p}$ consists of library function θ values taken at discussed time steps, from t_1 to t_k.

A standard approach to solve for the coefficients matrix would be the least square regression. However, it would generally generate $\boldsymbol{\xi}$ coefficient such that each element is nonzero, and therefore each function from the library θ would play a role in the EOM. The solution would result in a dense $\boldsymbol{\xi}$ matrix. Although minimizing the mean squared index, such a model might still predict the motion of the MBS inaccurately.

Another approach that raised recent interest in the field is to add an L^1 regularization term to the standard least square regression task

$$\boldsymbol{\xi}^* = \arg\min_{\boldsymbol{\xi}} \frac{1}{2}||\mathbf{U} - \boldsymbol{\Theta}(\mathbf{Y})\boldsymbol{\xi}||_2^2 + \alpha||\boldsymbol{\xi}||_1, \tag{10}$$

where α is a weight coefficient chosen to let the regression fit the linear coefficients acceptably but at the same time guarantee only a few elements of $\boldsymbol{\xi}$ matrix to be nonzero. Then, the resulting model has a limited number of functions from the function library activated, and consequently, it is possible to find a physical interpretation of identified equations of motion. Moreover, the obtained model is more general, less prone to overfitting, and tends to reproduce the nominal model more accurately.

The optimization task expressed in Eq. (10) can be solved with various numerical methods, one of which is the LASSO method (*least absolute shrinkage and selection operator*). In this work, we use a similar approach that originates from recently developed methods (*sequentially thresholded least squares*) [7], which were designed to eliminate near-zero elements of the coefficient matrix recursively and systematically.

4 The Adjoint Method in Optimal Control

Another matter that we investigated in this work is the optimal control of the MBS using the model obtained by the identification procedure described in Sect. 3. Once we have a system model, it is possible to implement a feedforward regulator to unburden the feedback loop and reduce its corrections to the control signal.

The proposed approach of solving the optimal control task with optimization methods lets one find the optimal control signals. What is more, it can also be considered a trajectory planner that accomplishes more abstract objectives, e.g., minimizing energy consumption, avoiding obstacles, or planning discontinuous events.

Let us express the task of finding the optimal control signal $\mathbf{u}^*(t)$ as minimization of the following cost function

$$\begin{cases} \mathbf{u}^*(t) = \arg\min_{\mathbf{u} \in \mathbb{U}} J(\mathbf{u}), \text{ where} \\ J(\mathbf{u}) = \int_0^T h(\mathbf{q},\dot{\mathbf{q}},\mathbf{u},t)\,dt + S(\mathbf{q},\dot{\mathbf{q}})|_{t=T}, \end{cases} \tag{11}$$

where integrand $h(\cdot)$ models cost in the whole time horizon and $S(\cdot)$ is the end time cost function.

The optimization task (11) is often solved with gradient methods, which in order to work effectively, need a method to compute the value of the gradient at a given point.

We can derive the analytical form of the discussed gradient as follows

$$\frac{\mathrm{d}J}{\mathrm{d}\mathbf{u}} = \int_0^T \left(\frac{\partial h}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial \mathbf{u}} + \frac{\partial h}{\partial \dot{\mathbf{q}}} \frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{u}} + \frac{\partial h}{\partial \mathbf{u}} \right) \mathrm{dt} + \left(\frac{\partial S}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial \mathbf{u}} \right)_{t=T} + \left(\frac{\partial S}{\partial \dot{\mathbf{q}}} \frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{u}} \right)_{t=T} . \tag{12}$$

Derivatives $\frac{\partial \mathbf{q}}{\partial \mathbf{u}}$ and $\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{u}}$ appear in gradient formulation (12). In general, computing these derivatives might become a difficult matter in the field of MBS due to the complex and nonlinear interrelations of variables enclosed in dynamics models. Nevertheless, the adjoint method [2, 11] helps and allows one to omit these complicated computations.

As the assumed linear model of MBS, apriori knowledge of functions library $\boldsymbol{\xi}$ and model (6) being an ordinary differential equation significantly simplify the derivatives problem, other methods, e.g., direct differentiation method [8], cannot be left without consideration. However, it must also be reminded that despite these particular features of the proposed formulation indicating a lack of necessity to implement the adjoint method, it can still become more efficient for large-scale systems.

Let us adjust the adjoint method to gradient computations of (11). First, we lower the order of (6) by introducing auxiliary variable $\mathbf{v}$ as

$$\left(\theta\left(\mathbf{q}, \mathbf{v}, \dot{\mathbf{v}}\right) \boldsymbol{\xi}\right)^T - \mathbf{u} = \mathbf{0}, \text{ where } \mathbf{v} = \dot{\mathbf{q}}. \tag{13}$$

Then, we extend cost function $J(\mathbf{u})$ by adding equations from (13)

$$\hat{J} = \int_0^T \left[h - \mathbf{d}^T \left(\left(\theta \boldsymbol{\xi}\right)^T - \mathbf{u} \right) - \mathbf{w}^T \left(\mathbf{v} - \dot{\mathbf{q}} \right) \right] \mathrm{dt} + S(\mathbf{q}, \mathbf{v})\big|_{t=T}, \tag{14}$$

where $\mathbf{d}, \mathbf{w} \in \mathbb{R}^n$ are Lagrange multipliers. Gradient of such an extended functional follows

$$\begin{aligned} \frac{\mathrm{d}\hat{J}}{\mathrm{d}\mathbf{u}} = \int_0^T \Bigg[& \frac{\partial h}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial \mathbf{u}} + \frac{\partial h}{\partial \mathbf{v}} \frac{\partial \mathbf{v}}{\partial \mathbf{u}} + \frac{\partial h}{\partial \mathbf{u}} - \mathbf{d}^T \frac{\partial}{\partial \mathbf{q}} \left(\theta \boldsymbol{\xi}\right)^T \frac{\partial \mathbf{q}}{\partial \mathbf{u}} - \mathbf{d}^T \frac{\partial}{\partial \mathbf{v}} \left(\theta \boldsymbol{\xi}\right)^T \frac{\partial \mathbf{v}}{\partial \mathbf{u}} + \\ & - \mathbf{d}^T \frac{\partial}{\partial \dot{\mathbf{v}}} \left(\theta \boldsymbol{\xi}\right)^T \frac{\partial \dot{\mathbf{v}}}{\partial \mathbf{u}} + \mathbf{d}^T - \mathbf{w}^T \frac{\partial \mathbf{v}}{\partial \mathbf{u}} + \mathbf{w}^T \frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{u}} \Bigg] \mathrm{dt} + \left. \frac{\partial S}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial \mathbf{u}} \right|_{t=T} + \left. \frac{\partial S}{\partial \mathbf{v}} \frac{\partial \mathbf{v}}{\partial \mathbf{u}} \right|_{t=T} . \end{aligned} \tag{15}$$

By applying integration by parts, we eliminate elements that include derivatives $\frac{\partial \dot{\mathbf{v}}}{\partial \mathbf{u}}$ and $\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{u}}$. Further, following the derivation shown in [11], we finally arrive at the following final condition problem

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{dt}} \mathbf{w} = \frac{\partial h}{\partial \mathbf{q}}^T - \frac{\partial}{\partial \mathbf{q}} \left(\theta \boldsymbol{\xi}\right) \mathbf{d}, \\ \frac{\mathrm{d}}{\mathrm{dt}} \left(\frac{\partial}{\partial \dot{\mathbf{v}}} \left(\theta \boldsymbol{\xi}\right) \mathbf{d} \right) = -\frac{\partial h}{\partial \mathbf{v}}^T + \mathbf{w} + \frac{\partial}{\partial \mathbf{v}} \left(\theta \boldsymbol{\xi}\right) \mathbf{d}, \\ \mathbf{w} = -\left. \frac{\partial S}{\partial \mathbf{q}}^T \right|_{t=T}, \quad \frac{\partial}{\partial \dot{\mathbf{v}}} \left(\theta \boldsymbol{\xi}\right) \mathbf{d} = \left. \frac{\partial S}{\partial \mathbf{v}}^T \right|_{t=T} . \end{cases} \tag{16}$$

If Lagrange multipliers are chosen in compliance with (16), the asked gradient of cost function $J(\mathbf{u})$ can be found by solving a simple integral

$$\frac{\mathrm{d}J}{\mathrm{d}\mathbf{u}} = \frac{\mathrm{d}\hat{J}}{\mathrm{d}\mathbf{u}} = \int_0^T \left(\frac{\partial h}{\partial \mathbf{u}} + \mathbf{d}^T \right) \mathrm{dt}. \tag{17}$$

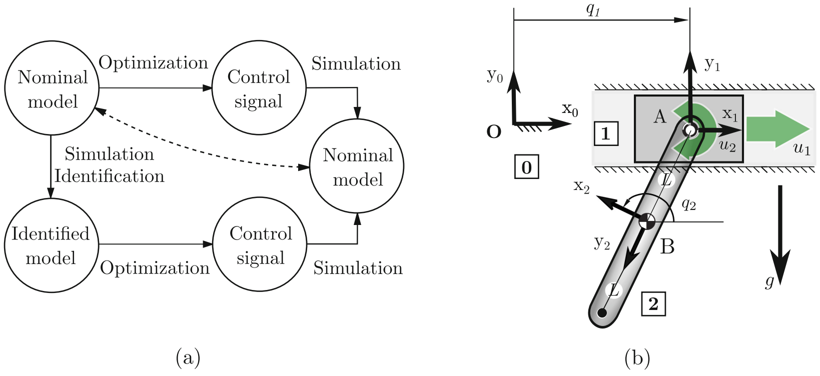


Fig. 2. (a) Test case scheme, (b) A cart on a pendulum

Summing up the adjoint method approach, it should be noted that a system of ordinary differential equations with final conditions (16) must be solved first before one can compute gradient (17). Moreover, in order to solve the final condition problem, complete knowledge of the system's state $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ in the whole time horizon $[0, T]$ must be known. It means that the optimization algorithm will sequentially solve, first, the forward dynamic problem (6) and then the adjoint equations in each iteration (16).

The need for multiple integrations along the time domain in consecutive iterations might affect computations time negatively. To help with this issue, one can reuse in (16) matrices from (6), which are already built and probably were also factorized. However, such an operation would require significant amounts of data to be written and read from memory, and the IO operations could become a time bottleneck. Nevertheless, a checkpointing approach can be implemented here [12]. This method is based on saving only a fraction of data parsimoniously scattered across the time domain and reintegrating only a chunk of the domain between checkpoints when needed. Ultimately, these methods can be accelerated by parallel computing on multicore and GPU-based machines.

5 Optimal Control of Planar Open-Chain MBS

The primary motivation for combining tools described in Sect. 3 and Sect. 4 was to investigate how inaccuracies in the model used for solving optimal control task affects the overall control results when the sought signals are applied to the real system. The real system in our setting is represented by a nominal model, so it is possible to conduct all the experiments in a virtual environment. A general overview of actions taken can be seen in the scheme presented in Fig. 2a.

For the purposes of this example, we choose a simple mechanism for simulations – a pendulum on a cart. Thanks to the choice, the nominal model of the MBS can be explicitly written in an elegant form. We record simulation data, i.e., measurements of position, velocity, and acceleration, as well as the control signals by feeding the system with sinusoidal controls. Then, we add noise to the saved measurements to imitate real sensor readings. Identification, represented by an arrow pointing from the nominal model to identified model in Fig. 2a, ends the first part of the experiment.

Next, we set a particular optimal control goal and design a proper cost function to be minimized. As shown in the scheme, two independent optimizations (with the adjoint method from Sect. 4 implemented) seeking the control signal are done. One has complete knowledge of the nominal mother, and the other uses the identified model only. It means that the optimizer works in ideal circumstances in the former setting, and the latter is more similar to a real-world case.

The final comparison is made by applying results from these two distinct optimizations to the nominal model. Such an approach lets one see differences in the control results and investigate how inaccuracies gained by the identification process affect the control quality.

Further discussion is divided into two parts. In Sect. 5.1, identification and its results are portrayed. Then, we solve optimization problems in two previously described scenarios and compare the results in Sect. 5.2.

The MBS of interest in this example is a pendulum on a cart placed in the gravitational field. The mechanism can be seen in Fig. 2b. We introduce joint coordinates q_1 and q_2. Both available degrees of freedom are controlled with signals u_1 and u_2, respectively, as indicated in the figure. In compliance with EOM form (6), the dynamics of such a system can be modeled by writing matrices $\mathscr{M}(\mathbf{q}, \dot{\mathbf{q}})$ and $\mathscr{F}(\mathbf{q}, \dot{\mathbf{q}})$ explicitly as follows

$$\begin{bmatrix} m_1 + m_2 & -m_2 L \cos q_2 \\ -m_2 L \cos q_2 & J_2 + m_2 L^2 \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} - \begin{bmatrix} -m_2 L \dot{q}_2^2 \sin q_2 \\ m_2 g L \sin q_2 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \tag{18}$$

where m_1, m_2 are masses of individual elements, and J_1, J_2 are their moments of inertia. For this test case, the physical parameters of the mechanism and environment are set as $L = 0.25\,\mathrm{m}$, $m_1 = 0.5\,\mathrm{kg}$, $m_2 = 0.2\,\mathrm{kg}$, $J_1 = J_2 = 0.005\ \mathrm{kgm}^2$, $g = 9.81\ \frac{m}{s^2}$.

We collect the data for identification from a simulation, which begins with initial conditions $q_1(0) = 0\,\mathrm{m}$, $q_2(0) = \frac{3\pi}{4}$ rad and feeds the control signals with $u_1(t) = \sin(0.01t)$, $u_2(t) = -0.01\sin(2t)$. Simulation time was set to 8 s. Collected measurements were distorted by the addition of noise with 10^{-6} amplitude and standard distribution.

5.1 Identification

Before starting the identification procedure described in Sect. 3, one must prepare function library $\boldsymbol{\theta}$. As we have the EOM of the model available explicitly (18), it is relatively easy to choose functions to insert into the library. The structure of Eq. (18) gives an idea that besides joint coordinates q_1, q_2 and their derivatives (velocity and acceleration), we should also include their nonlinear mappings, i.e., $\sin(q_1)$, $\cos(q_1)$, ..., $\cos(q_2)$. Ultimately, the library is a set generated by two-element combinations made from the following subset

$$\{1,\ q_1,\ \dot{q}_1,\ \ddot{q}_1,\ \sin(q_1),\ \cos(q_1),\ q_2,\ ...,\ \cos(q_2)\}. \tag{19}$$

Moreover, we add squared velocities $\dot{q}_1^2$, $\dot{q}_2^2 \sin(q_1)$, ..., $\dot{q}_2^2 \cos(q_2)$ to the library.

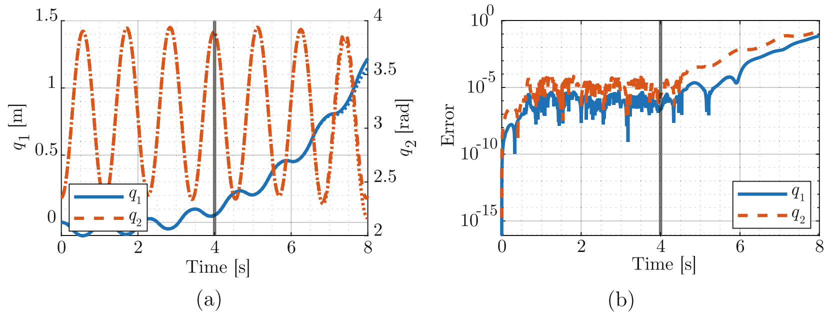


Fig. 3. Identification: (a) Position, (b) Error

For this case, we set regularization parameter $\alpha = 0.01$ from (10) and used data from the first 4 s of simulation to leave the other half for validation. In Fig. 3a, we present the joint coordinates time history taken from the simulation of the identified model overlaid on the data training and validation sets. The results diverge after time $t = 6$ s, 2 s after passing the training set domain. Therefore, we conclude that for this model, the prediction is more or less accurate for the 2 s time horizon. The identification error for both joint coordinates is shown in Fig. 3b. It can be seen that after 2 s of predicting, the error grows by 10^3 starting from approximately 10^{-6} at $t = 4$ s. Also, we can carefully conclude that prediction for bodies farther from the ground in the kinematic chain is worse than for the closer bodies.

Differences are also present in the structure of the identified mass matrix $\mathscr{M}^{ident}$

$$\mathscr{M}^{ident} = \begin{bmatrix} 0.10q_1 - 0.10\sin(q_1) + 0.70 & 0.01q_1 - 0.05\cos(q_2) - 0.09\sin(q_1) \\ 0.02q_1 - 0.05\cos(q_2) - 0.02\sin(q_1) & 0.18 \end{bmatrix} \quad (20)$$

compared to the nominal mass matrix $\mathscr{M}^{nom}$ from (18), which has the following form after inserting the parameters

$$\mathscr{M}^{nom} = \begin{bmatrix} 0.70 & -0.05\cos(q_2) \\ -0.05\cos(q_2) & 0.18 \end{bmatrix}. \quad (21)$$

Also, we can see a relationship between the weights of the functions that are common for both structures $\mathscr{M}^{ident}$ and $\mathscr{M}^{nom}$ - they are the same. Similar observations can be made for identified vector $\mathscr{F}$.

5.2 Optimal Control

The optimal control task for this MBS is to move the cart 1 m to the right in 1 s and simultaneously minimize the velocity. It can be translated into a mathematical form as a cost function.

$$J(\mathbf{u}) = \int_0^1 \left[10(q_1 - 1)^2 + 100\dot{q}_2^2\right] \mathrm{dt} + \left[100(q_1 - 1)^2 + 100\dot{q}_2^2\right]_{t=1}, \quad (22)$$

where the exact weights in the integrand and the end-time cost function were chosen arbitrarily, keeping in mind their reasonable ratio. The initial condition for the simulations was stable equilibrium ($q_1(0) = 0\,\mathrm{m}$, $q_2(0) = \pi\,\mathrm{rad}$). According to the scheme shown in Fig. 2a, two optimizations were run - one with the nominal model and one with the identified model. The optimizer used was a BFGS-based tool available in the *MATLAB* optimization package. All the differential equations were solved with the explicit Runge-Kutta method implemented in *MATLAB*'s solver *ode45* with absolute and relative tolerances set to $\mathrm{AbsTol} = \mathrm{RelTol} = 10^{-6}$.

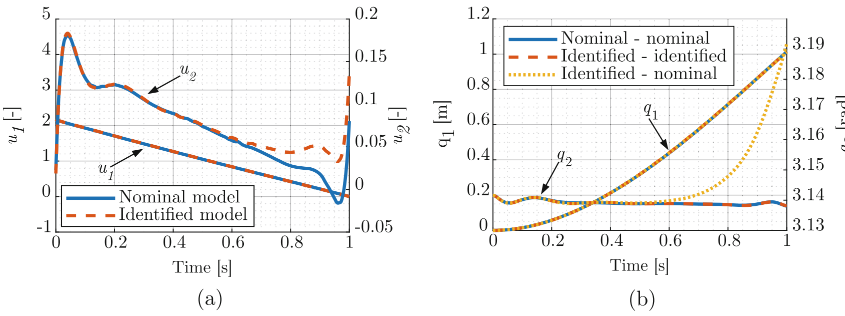


Fig. 4. (a) Comparison of control signals, (b) Comparison of coordinates

In both cases, the optimization method was stopped after 60 steps. In the scenario with the nominal model, the method decreased the cost from 109.97 to 4.86, and in the second scenario, the one with the nominal model, the cost decreased to 4.85. The resulting control signals are shown in Fig. 4a. Control signals u_1 related to the cart's horizontal movement overlap considerably for the whole time horizon, whereas the u_2 diverges significantly from time $t = 0.6\,\mathrm{s}$. It must be said that the extra functions present in the identified model have a substantial impact on the generated signals.

In Fig. 4b, results from simulations with different combinations of control signals and models are shown. Solid and dashed lines, which are results of feeding the control signals found with the nominal model to the nominal model (solid lines) and feeding the control signals found with the identified model to the identified model (dashed lines), are overlapped to a great extent. It means that in an ideal setting, the optimizer found signals that cause the same mechanism motion, an optimal one. The dotted line, which in the case of the second coordinate differs massively from the other paths, results from feeding the nominal model with the control signals found with the identified model.

The last case approximates the real-world scenario the best out of all three settings plotted. In that case, the cost computed from (22) is 22.40. Therefore, considerable sensitivity of the results to identified model's inaccuracies can be observed.

6 Summary and Conclusions

This work presents a set of methods that can be used for optimal control of MBS with models identified from data. Although these derivations are limited to planar and open-chain mechanisms, it does not affect the generality of the approach, which combine methods being an active field of research recently - methods of regression with regularization and identification and optimal control methods based on the adjoint method for computing gradients. The investigated MBS with two degrees of freedom indicates that the control quality found based on an identified model is notably sensitive to the quality of the identification results. The authors focus now on attenuating this inconvenience by including sensitivity to the identification's imperfections in the optimized cost function.

Acknowledgements. This work has been supported by National Science Center under grant No. 2018/29/B/ST8/00374.

References

1. Brunton, S., Proctor, J., Kutz, J.: Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc. Natl. Acad. Sci. U.S.A. **113**, 3932–3937 (2016). https://doi.org/10.1073/pnas.1517384113
2. Bryson, A.E., Ho, Y.C.: Applied Optimal Control. Hemisphere Publishing Corporation, Washington (1975)
3. Cao, Y., Li, S., Petzold, L., Serban, R.: Adjoint sensitivity analysis for differential-algebraic equations: the adjoint DAE system and its numerical solution. SIAM J. Sci. Comput. **24**, 1076–1089 (2003). https://doi.org/10.1137/S1064827501380630
4. Chadaj, K., Malczyk, P., Frączek, J.: A parallel Hamiltonian formulation for forward dynamics of closed-loop multibody systems. Multibody Syst. Dyn. **39**, 51–77 (2017). https://doi.org/10.1007/s11044-016-9531-x
5. Featherstone, R.: Rigid Body Dynamics Algorithms. Springer, Cham (2008). https://doi.org/10.1007/978-1-4899-7560-7
6. Hollerbach, J., Khalil, W., Gautier, M.: Model identification (Chapter 14). In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, pp. 113–138. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-32552-1_6
7. Kutz, J.N., Brunton, S.L.: Parsimony as the ultimate regularizer for physics-informed machine learning. Nonlinear Dyn. **107**, 1801–1817 (2022). https://doi.org/10.1007/s11071-021-07118-3
8. Maciąg, P., Malczyk, P., Frączek, J.: Hamiltonian direct differentiation and adjoint approaches for multibody system sensitivity analysis. Int. J. Numer. Methods Eng. **121**, 5082–5100 (2020). https://doi.org/10.1002/nme.6512
9. Malczyk, P., Frączek, J., González, F., Cuadrado, J.: Index-3 divide-and-conquer algorithm for efficient multibody system dynamics simulations: theory and parallel implementation. Nonlinear Dyn. **95**, 727–747 (2019). https://doi.org/10.1007/s11071-018-4593-3
10. Nachbagauer, K., Oberpeilsteiner, S., Sherif, K., Steiner, W.: The use of the adjoint method for solving typical optimization problems in multibody dynamics. J. Comput. Nonlinear Dyn. **10**, 061011 (2015). https://doi.org/10.1115/1.4028417
11. Pikuliński, M., Malczyk, P.: Adjoint method for optimal control of multibody systems in the Hamiltonian setting. Mech. Mach. Theory **166**, 104473 (2021). https://doi.org/10.1016/j.mechmachtheory.2021.104473
12. Serban, R., Hindmarsh, A.C.: CVODES: the sensitivity-enabled ODE solver in SUNDIALS. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, ASME, vol. 47438, pp. 257–269 (2005). https://doi.org/10.1115/DETC2005-85597

Minimum Energy Control of Robot Manipulators Using a Simple Variable Stiffness Mechanism

Makoto Iwamura(✉), Shunsuke Ide, Yeongju Baek, Yoshiki Maeda, and Kodai Ryomoto

Department of Mechanical Engineering, Fukuoka University, 19-1, Nanakuma 8-Chome, Jonan-ku, Fukuoka 814-0180, Japan
iwamura@fukuoka-u.ac.jp

Abstract. This paper proposes a concept for the design and control of an energy saving manipulator utilizing passive elastic elements for energy storage. Firstly, we review our previously proposed method and the practical design of an energy saving manipulator briefly. This method reduces the energy consumption by using natural frequencies and eigenmodes of the system. Therefore, when the weights of the system are changed by transferring some objects or when the boundary conditions are changed, we should adjust stiffnesses of the system. Hence, in this paper, we propose a very simple variable stiffness mechanism that can change rotational stiffnesses around the manipulator joint axes. This mechanism makes it possible to adjust the natural frequencies and eigenmodes of the manipulator. A prototype 2DOF manipulator with the variable stiffness mechanism is developed by using the linear springs and the reaction wheels to verify the proposed method. Experimental results show the effectiveness of the proposed energy saving manipulator concept.

1 Introduction

Toward the realization of carbon neutrality, thorough energy conservation measures are being considered in various fields. Since many robots used in factories also consume a huge amount of energy every day, their energy saving is an important issue. Therefore, research on energy saving of robots has been actively conducted so far [1,2]. The authors have studied an energy-saving control method for robots using passive storage elements such as springs [3]. In the proposed method, the energy consumption is reduced by adding springs to the joints of the manipulator and using the eigenfrequency and eigenmode of the link system. However, when the mass of the system changes due to the grasping of the object, or when the boundary conditions at the start and end points are changed, it is necessary to readjust the eigenfrequency and eigenmode of the system. Therefore, in this paper, we propose a simple variable stiffness mechanism that can be installed at the joint of the manipulator. Using the method of multibody dynamics, we derive the spring constant conversion formula and formulate the stiffness adjustment rule based on it. In addition, a planar 2DOF manipulator equipped with the proposed variable stiffness mechanism is actually manufactured, and the validity of the derived

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 110–119, 2024.
https://doi.org/10.1007/978-3-031-50000-8_10

theoretical formula is verified. Finally, we confirm that it becomes possible to flexibly change the boundary conditions by using the proposed variable stiffness mechanism.

2 Design and Control of Energy-Saving Manipulator

Figure 1 shows the configuration of the proposed energy-saving manipulator. In the proposed method, the joints must be able to rotate freely in order to use the natural vibration of the system, and motors cannot be installed in the joints like a normal manipulator. Therefore, the joints are free joints, and the torque is applied from the reaction wheels installed at the appropriate positions of the links. In addition, a mechanism consisting of two tension springs and a special spring holder (see Chap. 3 for details) is installed in the joint to give rotational stiffness. The equations of motion for the link and the reaction wheel can be expressed as follows.

$$\boldsymbol{M}_{\theta\theta}\ddot{\boldsymbol{\theta}} + \boldsymbol{M}_{\phi\theta}^T\ddot{\boldsymbol{\phi}} + \boldsymbol{h} = -\boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{\theta}_n) \tag{1}$$

$$\boldsymbol{M}_{\phi\theta}\ddot{\boldsymbol{\theta}} + \boldsymbol{M}_{\phi\phi}\ddot{\boldsymbol{\phi}} = \boldsymbol{\tau} \tag{2}$$

where $\boldsymbol{\theta} = [\theta_1, \theta_2 \cdots, \theta_N]^T$ is the vector of joint variables, $\boldsymbol{\phi} = [\phi_1, \phi_2 \cdots, \phi_N]^T$ is the vector of rotation angles of reaction wheels, $\boldsymbol{M}_{\theta\theta}$, $\boldsymbol{M}_{\phi\theta}$, $\boldsymbol{M}_{\phi\phi}$ are the inertia matrices, $\boldsymbol{h}$ is the vector of centrifugal and Coriolis forces, $\boldsymbol{\tau} = [\tau_1, \tau_2 \cdots, \tau_N]^T$ is the vector of driving torques of reaction wheels, $\boldsymbol{K} = \mathrm{diag}[k_1, k_2, \ldots, k_N]$ is the stiffness matrix (k_i is the equivalent spring constant obtained by converting the stiffness of the two tension springs into the rotational stiffness around the joint axis), $\boldsymbol{\theta}_n$ is the vector representing the mounting angle at which the spring has a natural length.

Eliminating $\boldsymbol{\phi}$ from Eqs. (1) and (2) yields the following equations of motion

$$\boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{h}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = -\boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{\theta}_n) + \boldsymbol{u} \tag{3}$$

where $\boldsymbol{M} \equiv \boldsymbol{M}_{\theta\theta} - \boldsymbol{M}_{\phi\theta}^T\boldsymbol{M}_{\phi\phi}^{-1}\boldsymbol{M}_{\phi\theta}$, $\boldsymbol{u} \equiv -\boldsymbol{M}_{\phi\theta}^T\boldsymbol{M}_{\phi\phi}^{-1}\boldsymbol{\tau}$.

Here, we consider a motion that stops at $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_0$ and stops again at $\boldsymbol{\theta}(t_f) = \boldsymbol{\theta}_f$. Energy consumption is evaluated using the following cost function.

$$J = \int_0^{t_f} f_0(\boldsymbol{x}, \boldsymbol{u})dt,\ f_0(\boldsymbol{x}, \boldsymbol{u}) = \frac{1}{2}\boldsymbol{u}^T\boldsymbol{W}\boldsymbol{u} \tag{4}$$

where $\boldsymbol{W} \in R^{N\times N}$ is a positive definite symmetric matrix. An energy-saving effect can be obtained by appropriately selecting the spring parameters and motion trajectory. In the following, the control input $\boldsymbol{u}(t)$, trajectory $\boldsymbol{\theta}(t)$, spring constant $\boldsymbol{k} = [k_1, k_2, \cdots, k_N]^T$, and spring mounting angle $\boldsymbol{\theta}_n$ that minimize the cost function (4) will be explained.

Since Eq. (3) is nonlinear and it is difficult to obtain an analytical solution, here we linearize the equations of motion to obtain an approximate solution and analyze the basic characteristics of the optimal solution. First, the coordinate reference point is shifted to the middle point between the initial state and the final state in order to make the boundary conditions symmetrical. That is, define $\boldsymbol{\theta}_m = \frac{1}{2}(\boldsymbol{\theta}_f + \boldsymbol{\theta}_0)$, $\boldsymbol{\theta}_e = \frac{1}{2}(\boldsymbol{\theta}_f - \boldsymbol{\theta}_0)$ and transform the coordinates as $\tilde{\boldsymbol{\theta}}(t) = \boldsymbol{\theta}(t) - \boldsymbol{\theta}_m$, $\tilde{\boldsymbol{\theta}}_n = \boldsymbol{\theta}_n - \boldsymbol{\theta}_m$. Then,

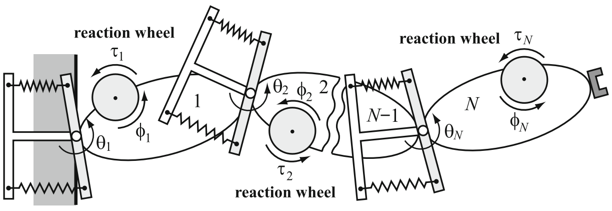


Fig. 1. Structure of proposed energy saving manipulator

approximate the inertia matrix at the midpoint $\tilde{\boldsymbol{\theta}} = \mathbf{0}$ ($\boldsymbol{\theta} = \boldsymbol{\theta}_m$) as $\boldsymbol{M}(\boldsymbol{\theta}_m) = \hat{\boldsymbol{M}}$, and neglect the centrifugal and Coriolis forces $\boldsymbol{h}$, the equations of motion are linearized as follows.

$$\hat{\boldsymbol{M}}\ddot{\tilde{\boldsymbol{\theta}}} + \boldsymbol{K}\tilde{\boldsymbol{\theta}} = \boldsymbol{u} + \boldsymbol{K}\tilde{\boldsymbol{\theta}}_n \tag{5}$$

Let us consider the free vibration system and calculate the modal matrix $\boldsymbol{\Phi} \in R^{N\times N}$ that satisfies $\boldsymbol{\Phi}^T\hat{\boldsymbol{M}}\boldsymbol{\Phi} = \boldsymbol{I}$, $\boldsymbol{\Phi}^T\boldsymbol{K}\boldsymbol{\Phi} = \boldsymbol{\Omega}^2$, $\boldsymbol{\Omega} = \mathrm{diag}[\omega_1, \omega_2, \cdots, \omega_N]$, where $\boldsymbol{I}$ is the identity matrix and ω_i is the i-th natural frequency. We make the coordinate transformation $\boldsymbol{q} = \boldsymbol{\Phi}^{-1}\tilde{\boldsymbol{\theta}}$ ($\boldsymbol{q}_n = \boldsymbol{\Phi}^{-1}\tilde{\boldsymbol{\theta}}_n$) and define the state vectors $\boldsymbol{x} = [\boldsymbol{x}_1^T, \boldsymbol{x}_2^T]^T = [\boldsymbol{q}^T, \dot{\boldsymbol{q}}^T]^T$. Then, the following state equations are found.

$$\dot{\boldsymbol{x}}_1 = \boldsymbol{x}_2 \ (\equiv \boldsymbol{f}_1(\boldsymbol{x}, \boldsymbol{u})) \tag{6}$$
$$\dot{\boldsymbol{x}}_2 = -\boldsymbol{\Omega}^2\boldsymbol{x}_1 + \boldsymbol{\Phi}^T\boldsymbol{u} + \boldsymbol{\Omega}^2\boldsymbol{q}_n \ (\equiv \boldsymbol{f}_2(\boldsymbol{x}, \boldsymbol{u})) \tag{7}$$

Next, let us introduce an adjoint vector $\boldsymbol{\psi} = [\boldsymbol{\psi}_1^T, \boldsymbol{\psi}_2^T]^T$ and define the Hamiltonian as $H = f_0 + \boldsymbol{\psi}_1^T\boldsymbol{f}_1 + \boldsymbol{\psi}_2^T\boldsymbol{f}_2$. Then, the optimal control is derived from the condition $\partial H/\partial\boldsymbol{u} = \mathbf{0}$ as $\boldsymbol{u} = -\boldsymbol{W}^{-1}\boldsymbol{\Phi}\boldsymbol{\psi}_2$. Substituting this into H, the Hamiltonian along the optimal trajectory is given by

$$H = \boldsymbol{\psi}_1^T\boldsymbol{x}_2 - \boldsymbol{\psi}_2^T\boldsymbol{\Omega}^2\boldsymbol{x}_1 - \frac{1}{2}\boldsymbol{\psi}_2^T\boldsymbol{\Phi}^T\boldsymbol{W}^{-1}\boldsymbol{\Phi}\boldsymbol{\psi}_2 + \boldsymbol{\psi}_2^T\boldsymbol{\Omega}^2\boldsymbol{q}_n \tag{8}$$

From Eq. (8), the canonical equations of Hamilton can be derived as follows.

$$\dot{\boldsymbol{x}} = \partial H/\partial\boldsymbol{\psi} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{\psi} + \boldsymbol{c}_n \tag{9}$$
$$\dot{\boldsymbol{\psi}} = -\partial H/\partial\boldsymbol{x} = -\boldsymbol{A}^T\boldsymbol{\psi} \tag{10}$$

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ -\boldsymbol{\Omega}^2 & \mathbf{0} \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\boldsymbol{\Phi}^T\boldsymbol{W}^{-1}\boldsymbol{\Phi} \end{bmatrix}, \boldsymbol{c}_n = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\Omega}^2\boldsymbol{q}_n \end{bmatrix}$$

By solving the differential Eqs. (9) and (10) under the boundary conditions $\boldsymbol{x}(0) = [-\boldsymbol{q}_e^T, \mathbf{0}^T]^T$ and $\boldsymbol{x}(t_f) = [\boldsymbol{q}_e^T, \mathbf{0}^T]^T$ ($\boldsymbol{q}_e = \boldsymbol{\Phi}^{-1}\boldsymbol{\theta}_e$), we obtain the optimal solution that minimizes the energy consumption. Choosing the weighting matrix as $\boldsymbol{W} = \hat{\boldsymbol{M}}^{-1}$ allows to decouple the equations by the property $\boldsymbol{\Phi}^T\boldsymbol{W}^{-1}\boldsymbol{\Phi} = \boldsymbol{\Phi}^T\hat{\boldsymbol{M}}\boldsymbol{\Phi} = \boldsymbol{I}$. In this case, we can solve the problem analytically and the main results can be summarized as follows.

The optimal spring mounting angle that minimizes J is always $\boldsymbol{q}_n = \mathbf{0}$ ($\boldsymbol{\theta}_n = \boldsymbol{\theta}_m$). The optimal control $\boldsymbol{u}(t)$ can be computed by

$$\boldsymbol{u}(t) = -\hat{\boldsymbol{M}}\boldsymbol{\Phi}\boldsymbol{\psi}_2(t),\ \boldsymbol{\psi}_2(t) = [\psi_{21}(t), \psi_{22}(t), \cdots, \psi_{2N}(t)]^T$$
$$\psi_{2i}(t) = \frac{2\omega_i^2\{\sin\omega_i(t_f - t) - \sin\omega_i t\}}{\sin\omega_i t_f - \omega_i t_f} q_{ei} \tag{11}$$

The optimal trajectory can be expressed as

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}_m + \boldsymbol{\Phi}\boldsymbol{q}(t),\ \boldsymbol{q}(t) = \boldsymbol{x}_1(t) = [q_1(t), q_2(t), \cdots, q_N(t)]^T$$
$$q_i(t) = -q_{ei}\left\{\cos\omega_i t + \frac{\omega_i t\sin\omega_i t\sin\omega_i t_f}{\sin\omega_i t_f - \omega_i t_f} + \frac{(\omega_i t\cos\omega_i t - \sin\omega_i t)(1+\cos\omega_i t_f)}{\sin\omega_i t_f - \omega_i t_f}\right\} \tag{12}$$

The relationship between the minimum value of the energy consumption J and the robot cycle time t_f can be summarized as follows

$$J(t_f) = \sum_{i=1}^{N} J_i(t_f),\ J_i(t_f) = \begin{cases} \dfrac{2\omega_i^3(1+\cos\omega_i t_f)}{\omega_i t_f - \sin\omega_i t_f} q_{ei}^2 & (\text{ if } \omega_i \neq 0) \\ \dfrac{24}{t_f^3} q_{ei}^2 & (\text{ if } \omega_i = 0) \end{cases} \tag{13}$$

These theoretical solutions were derived based on the linearized equations of motion. However, using a special numerical calculation method for optimal control [4], we have confirmed that these approximate solutions closely approximate the basic properties of the exact solutions.

If we consider t_f as the quantity that may take any value, $H(t_f) = 0$ should be satisfied from the transversality condition. Moreover, since Eq. (8) does not contain t explicitly, $\partial H/\partial t = 0$, it holds H=const. along optimal trajectories. Hence the condition $H(0) = 0$ should be satisfied. By substituting $\boldsymbol{x}(0) = [\boldsymbol{x}_1(0)^T, \boldsymbol{x}_2(0)^T]^T = [-\boldsymbol{q}_e^T, \mathbf{0}^T]^T$ and $\boldsymbol{\psi}_2$ into Eq. (8), one can get

$$H(0) = \sum_{i=1}^{N} \frac{-2\omega_i^5 t_f \sin\omega_i t_f}{(\omega_i t_f - \sin\omega_i t_f)^2} q_{ei}^2 \tag{14}$$

From Eq. (14), it follows that $H(0) = 0$ is satisfied if $\sin\omega_i t_f = 0$ ($i = 1, 2, \cdots, N$) or equivalently $\omega_i t_f = r_i\pi$ ($i = 1, 2, \cdots, N$), where r_i is an integer. When the condition $\omega_i t_f = r_i\pi$ is satisfied, the cost function (13) can be expressed as

$$J = \sum_{i=1}^{N} J_i = \sum_{i=1}^{N} \frac{2\omega_i^3(1+\cos r_i\pi)}{r_i\pi} q_{ei}^2 \tag{15}$$

From Eq. (15), it is understood that J_i takes the maximum $4\omega_i^3 q_{ei}^2/r_i\pi$ if r_i is an even number and vanishes if r_i is an odd numbers. Hence, J takes the global minimum if all r_i are odd number resulting in the minimum value of zero.

Next, we consider the problem to design the spring stiffnesses $\boldsymbol{k} = [k_1, k_2, \cdots, k_N]^T$ that make the consumed energy minimum for a specified time t_f^*. Firstly, from the optimal conditions $\omega_i t_f = r_i\pi$, the natural frequencies read as $\omega_i = r_i\pi/t_f^*$ ($i = 1, 2, \cdots, N$), where all r_i should be selected to be odd numbers so that all J_i takes the minimum. The spring stiffnesses $\boldsymbol{k} = [k_1, k_2, \cdots, k_N]^T$ should be determined as they satisfy the characteristic equations $\det[\boldsymbol{K} - \omega_i^2 \hat{\boldsymbol{M}}] = 0$ $(i = 1, 2, \cdots, N)$. Let us define the error vector $\boldsymbol{e} = [e_1, e_2, \cdots, e_N]^T$ where $e_i = \det[\boldsymbol{K} - \omega_i^2 \overline{\boldsymbol{M}}]$. Then, the problem here becomes to find $\boldsymbol{k}$ that satisfies $\boldsymbol{e}(\boldsymbol{k}) = \boldsymbol{0}$. Solving this nonlinear equation, e.g., by Newton-Raphson method, we can obtain the optimal spring stiffnesses $\boldsymbol{k}$ that minimizes the energy consumption. Finally, we can achieve the minimum energy control of planar robot manipulators by adding the springs with optimal stiffnesses to the joint at the optimal mounting angles $\boldsymbol{\theta}_m$.

3 Variable Stiffness Mechanism

As explained in the previous chapter, the optimum spring constant $\boldsymbol{k} = [k_1, k_2, \cdots, k_N]^T$ is calculated from the characteristic equations $\det[\boldsymbol{K} - \omega_i^2 \hat{\boldsymbol{M}}] = 0$ $(i = 1, 2, \cdots, N)$. Since $\hat{\boldsymbol{M}} = \boldsymbol{M}(\boldsymbol{\theta}_m)$, $\boldsymbol{\theta}_m = \frac{1}{2}(\boldsymbol{\theta}_f + \boldsymbol{\theta}_0)$, the optimum spring constant changes if the boundary conditions $\boldsymbol{\theta}_0, \boldsymbol{\theta}_f$ are changed or if the mass included in $\boldsymbol{M}$ changes when the manipulator grips and transports the object. Since it is not practical to replace springs every time the boundary conditions or mass change, a mechanism that can adjust the spring constant is required. Therefore, in this chapter, we propose a simple variable stiffness mechanism that can be added to the joints of the manipulator.

Figure 2 shows the proposed variable stiffness mechanism for one axis. For simplicity of notation, the subscript i that was added to indicate the joint axis number in the previous chapter is omitted. The rod-shaped part Ⓐ is rigidly connected to the child link on the tip side, and the T-shaped part Ⓑ is fixed to the parent link on the root side at the optimum mounting angle θ_n. Here, we consider changing the stiffness by sliding the mounting positions S_1, S_2 of the tension springs by the amount b. In the following, we denote the distance from the rotation axis to the spring mounting points P_1, P_2 as a, the spring constant of the tension spring attached between P_jS_j as k_t, and the natural length as l_0. Σ is a coordinate system in which the origin is set on the rotation axis and the y-axis is aligned with the longitudinal direction of the parent link. Σ_A and Σ_B are coordinate systems fixed to parts A and B, respectively, as shown in the Fig. 2.

Firstly, we calculate the torque that this mechanism produces around the point O when the child link rotates θ with respect to the parent link. The vector $\boldsymbol{d}_j$, which represents the vector from the point S_j to the point P_j in the Σ coordinate system, can be calculated as follows.

$$\boldsymbol{d}_j = \boldsymbol{r}_j^P - \boldsymbol{r}_j^S = \boldsymbol{A}\overline{\boldsymbol{u}}_j^P - \boldsymbol{B}(\overline{\boldsymbol{u}}_j^S - \overline{\boldsymbol{u}}^O) \tag{16}$$

where, $\overline{\boldsymbol{u}}_j^P$ is the vector from the point O to the point P_j expressed in the Σ_A, $\overline{\boldsymbol{u}}_j^S$ and $\overline{\boldsymbol{u}}^O$ are the vectors from the origin of Σ_B to the points S_j and O expressed in the Σ_B. Moreover, $\boldsymbol{A}$ is the rotation matrix from Σ_A to Σ, $\boldsymbol{B}$ is the rotation matrix from Σ_B to Σ, and they can be expressed as follows.

$$\boldsymbol{A} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} \cos\theta_n & -\sin\theta_n \\ \sin\theta_n & \cos\theta_n \end{bmatrix} \tag{17}$$

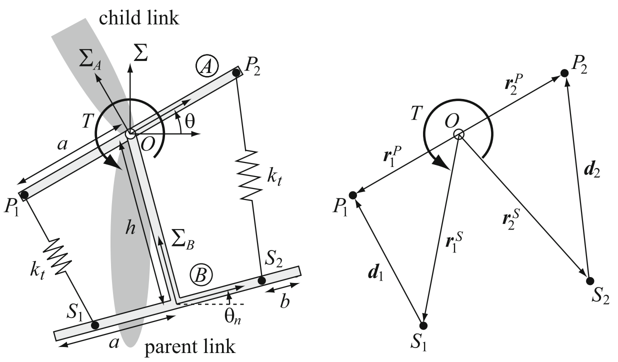


Fig. 2. Simple Variable Stiffness Mechanism

Since the total length of the tension spring can be calculated by $\|\boldsymbol{d}_j\|$, the torque Q_j produced by one tension spring around the point O can be calculated as follows.

$$Q_j(\theta) = -k_t(\|\boldsymbol{d}_j\| - l_0)(\boldsymbol{A}\boldsymbol{V}\bar{\boldsymbol{u}}_j^P)^T \frac{\boldsymbol{d}_j}{\|\boldsymbol{d}_j\|} \tag{18}$$

where $\boldsymbol{V}$ is a rotation matrix that rotates the vector counterclockwise by 90°. Then, the torque T induced around the point O by two tension springs can be calculated as

$$T(\theta) = Q_1(\theta) + Q_2(\theta) \tag{19}$$

Next, we derive an equation for calculating the equivalent rotational spring constant k around the joint axis from the spring constant k_t of the tension spring. As explained in Chap. 2, the optimal spring mounting angle is $\boldsymbol{\theta}_n = \boldsymbol{\theta}_m = \frac{1}{2}(\boldsymbol{\theta}_0 + \boldsymbol{\theta}_f)$. Therefore, without loss of generality, we can assume $\boldsymbol{\theta}_n = \boldsymbol{0}$, since the spring will expand and contract symmetrically about the equilibrium point. Hence, we can derive the spring constant conversion formula by considering the motion around $\boldsymbol{\theta} = \boldsymbol{0}$. When $\boldsymbol{\theta}_n = \boldsymbol{0}$, Eq. (19) can be calculated as follows.

$$\begin{aligned} T(\theta) = &-k_t\left(1 - \frac{l_0}{\|\boldsymbol{d}_1(\theta)\|}\right)\{a(a-b)\sin\theta + ah\cos\theta\} \\ &-k_t\left(1 - \frac{l_0}{\|\boldsymbol{d}_2(\theta)\|}\right)\{a(a-b)\sin\theta - ah\cos\theta\} \end{aligned} \tag{20}$$

Equation 20) is Taylor-expanded around $\theta = 0$, and after first-order approximation, it is considered to be balanced with the torque $-k\theta$ around the point O, that is

$$T(\theta) \cong T(0) + \frac{\partial T(0)}{\partial \theta}\theta \equiv -k\theta \tag{21}$$

When $\theta = 0$, $\|\boldsymbol{d}_1(0)\| = \|\boldsymbol{d}_2(0)\| = \sqrt{h^2 + b^2}$, so the first term in the middle side of Eq. (21) becomes 0. Calculate the derivative $\partial T(0)/\partial\theta$ of the second term of the middle side of Eq. (21), and compare the middle side with the right side. Then, the following conversion formula for equivalent rotational spring constant k and tension spring constant k_t is obtained.

$$k = 2k_t \left\{ \frac{l_0 a^2 h^2}{(\sqrt{h^2+b^2})^3} + \left(1 - \frac{l_0}{\sqrt{h^2+b^2}}\right) a(a-b) \right\} \tag{22}$$

Finally, we consider how to obtain the sliding amount b of the spring mounting point, when the desired rotational spring constant k^* is specified. From Eq. (22), k can be considered as a function of b, so it is expressed as $k(b)$. Then, the problem is to find b that satisfies the nonlinear equation $f(b) = k(b) - k^* = 0$. To solve this problem numerically using Newton-Raphson method, the iteration formula is

$$b^{(\alpha+1)} = b^{(\alpha)} - \frac{f(b^{(\alpha)})}{f'(b^{(\alpha)})} \tag{23}$$

where α is the iteration number. When using Newton-Raphson method, it is important to select the initial value $b^{(0)}$ appropriately. If $k(b)$ is Taylor-expanded around $b = 0$ and first-order approximation is used, the following equation is obtained.

$$k(b) \cong k(0) + \frac{\partial k(0)}{\partial b} b = 2k_t a^2 \left\{ 1 - \left(1 - \frac{l_0}{h}\right)\left(\frac{b}{a}\right) \right\} \tag{24}$$

Solving the above equation for b gives the following equation.

$$b = \frac{(2k_t a^2 - k)h}{2k_t a(h - l_0)} \tag{25}$$

By substituting the desired optimum spring constant k^* into k on the right hand side of Eq. (25), the appropriate initial value $b^{(0)}$ when applying the Newton-Raphson method can be calculated.

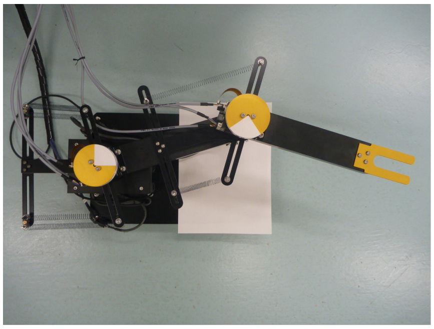


Fig. 3. Picture of energy saving manipulator

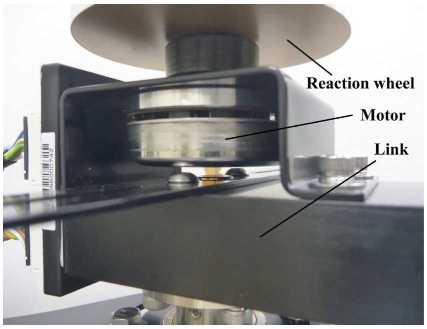


Fig. 4. Enlarged view of the joint mechanism

4 Experimental Verification

A variable stiffness mechanism proposed in Chap. 3 and an energy-saving manipulator equipped with it were prototyped. Figures 3 and 4 show the appearance of the experimental setup. A DC motor is used to drive the reaction wheel, and a rotary encoder is installed at the joint to measure the rotation angle of the link. In addition, an electromagnetic brake is installed in the joint so that the link can be stopped at any angle.

Firstly, the validity and accuracy of the spring constant conversion formula (22) derived in Chap. 3 are verified using the second link of the experimental device. Push a point on the 2nd link at a distance l away from the 2nd joint with a force gauge perpendicular to the link, and measure the force F required to tilt the 2nd link by θ. Here, this is repeated 5 times and the average value F_{ave} is obtained, and the torque T acting on the rotating axis is calculated by $T = F_{ave}l$. As an example, when $b = 0$[m], θ is changed by 1[deg] from 2[deg] to 15[deg]. Figure 5 shows the result of calculating the torque T with circles. The solid line in the figure is the approximate straight line that passes through the origin using the method of least squares from the obtained data. If the equivalent rotational spring constant is k, there is a relationship of $T = k\theta$, so $k = 0.0016$[Nm/rad] can be obtained from the slope of the approximated straight line. Similar measurements and calculations are performed for b from 0[m] to 0.07[m] by 0.01[m]. Plotting the obtained k against b results in the circles in Fig. 6. The solid line in the figure is the result of calculation using the theoretical formula (22). Both results are in good agreement, which confirms the validity of Eq. (22).

Next, we investigate the adjustable range of k by the proposed variable stiffness mechanism. Figure 7 shows the result of calculating how much the rotational spring constant k can be changed by changing the movement amount b of the spring mounting point based on Eq. (22). For example, when using a tension spring with a spring constant of $k_t = 10.0$[N/m], by changing b from 0[m] to 0.1[m], spring constant k can be changed by about 8 times. Also, it can be seen that the larger the value of k_t, the wider the range of k that can be changed.

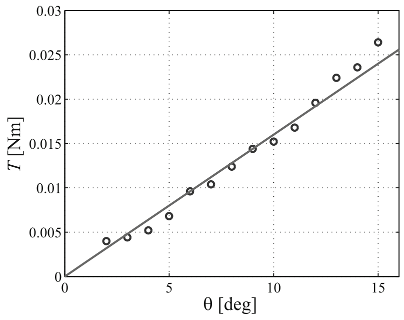


Fig. 5. Relationship between θ and T.

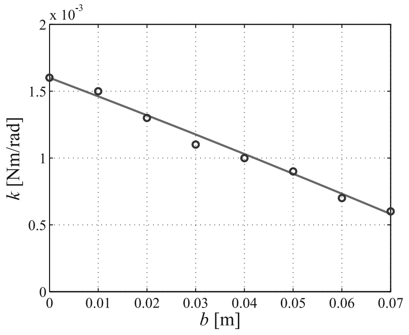


Fig. 6. Relationship between b and k.

Finally, it is verified whether it is possible to move between specified points by adjusting the mounting points of the springs even when the boundary conditions are changed by using the proposed variable stiffness mechanism. Here, adaptive control is used for trajectory tracking control. The motion time is set to $t_f^* = 2$[s], and we first consider the case where the robot stops at $\boldsymbol{\theta}_0 = [0,0]^T$[deg] and then stops again at $\boldsymbol{\theta}_f = [40,40]^T$[deg]. Obtaining the desired spring constant by the optimal spring design method explained in Chap. 2, and calculating the spring mounting point movement amount to achieve that spring constant by the method proposed in Chap. 3, result in $\boldsymbol{b} = [b_1, b_2]^T = [0.000, 0.079]^T$[m]. Figure 8 shows the experimental result when the spring mounting points were adjusted to this position. In the figure, the desired trajectory of the end effector's attitude angle ϕ is indicated by a dotted line, and the actual trajectory is indicated by a solid line. From this, it can be confirmed that the target of 80[deg] is reached at the final time. Next, Fig. 9 shows the experimental results when the target point is changed to $\boldsymbol{\theta}_f = [50, 50]^T$[deg] without changing the spring constant. From the figure, it can be confirmed that there is an error of about 10[deg] at the final time. Therefore, based on the changed boundary conditions, the optimal spring constant is calculated again by the method in Chap. 2, and the appropriate spring mounting point movement is calculated by the method proposed in Chap. 3, result in $\boldsymbol{b} = [b_1, b_2]^T = [0.006, 0.069]^T$[m]. Figure 10 shows the experimental results when the spring mounting point was adjusted to this position. From the figure, it can be confirmed that the target of 100[deg] is reached at the final time. From the above, it was confirmed that the proposed variable stiffness mechanism can flexibly cope with different boundary conditions. We have confirmed that the proposed method can reduce energy consumption by about 95% compared to conventional manipulators.

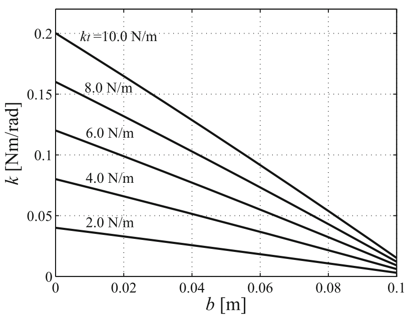


Fig. 7. Variable range of k

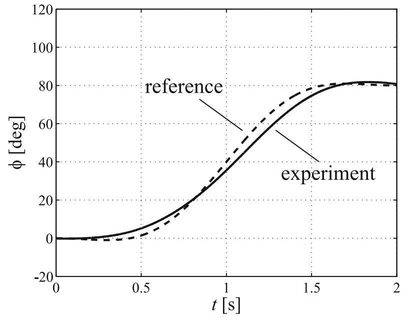


Fig. 8. Tip angle ($\boldsymbol{\theta}_f = [40, 40]^T$deg)

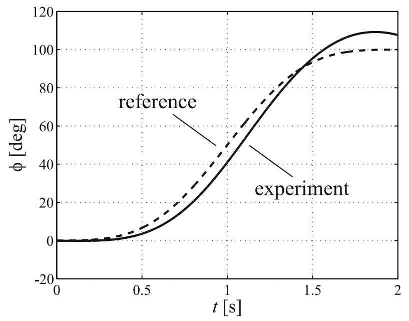


Fig. 9. Tip angle ($\boldsymbol{\theta}_f = [50, 50]^T$deg, before adjust stiffnesses)

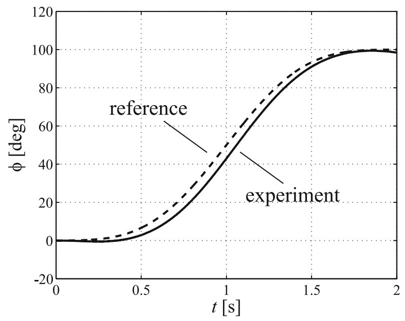


Fig. 10. Tip angle ($\boldsymbol{\theta}_f = [50, 50]^T$deg, after adjust stiffnesses)

5 Conclusions

In this study, we investigated a method of driving the manipulator with significantly less energy consumption than usual by adding springs to the joints and effectively using the elastic potential energy. We developed a simple variable stiffness mechanism that can be attached to a joint, and demonstrated that it can flexibly respond to changes in the manipulator's mass and boundary conditions.

References

1. Schiehlen, W., Guse, N.: Powersaving control of mechanisms. In: Vibration control of nonlinear mechanisms and structures (IUTAM Symposium, July 18-22 2005, Munich, Germany). Ulbrich, H., Gunthner, W. (eds.), pp. 277–286, Springer (2005). https://doi.org/10.1007/1-4020-4161-6_25
2. Meike, D., Ribickis, L.: Recuperated energy savings potential and approaches in industrial robotics. In: IEEE International Conference on Automation Science and Engineering (2011)
3. Iwamura, M., Schiehlen, W.: Minimum control energy in multibody systems using gravity and springs. J. Syst. Design Dyn. **5**(3), 474–485 (2011)
4. Iwamura, M., Eberhard, P., Schiehlen, W., Seifried, R.: A general purpose optimal trajectory planning algorithm for multibody systems. In: Proceedings of the 28th IASRED International Conference on Modeling, Identification, and Control, Hangos, K.M. (ed.), pp. 55–62 (2009)

Control Design of Machine Tool Feed Drives Using Mechatronic System Simulation

Roman Binder[1(✉)], Katrin Ellermann[2], and Harald Sehrschön[1]

[1] Fill Gesellschaft m.b.H., Research and Development, Fillstrasse 1, 4942 Gurten, Austria
{roman.binder,harald.sehrschoen}@fill.co.at

[2] Graz University of Technology, Institue of Mechanics, Kopernikusgasse 24/IV, 8010 Graz, Austria
ellermann@tugraz.at

Abstract. Machine tools are among the most important components in modern production engineering where cost-effective manufacturing of parts with high geometric accuracy is required. The mechanical components of the machine together with feed drives and digital control loops form a complex mechatronic system which must be designed and optimized simultaneously. A mechatronic system simulation is developed, suitable for mutual optimization of mechanical structure, feed drives and control loops including multi-axis configurations like gantry and main-sub. Simulation results are compared to measurements on real machine tools and show excellent agreement. Furthermore, flatness-based exact feed forward control is used to significantly improve machine tool performance in simulation studies.

1 Introduction

The demand for high accuracy of finished workpieces within short cycle times requires a high static and dynamic structural stiffness of machine tools to ensure low path deflection during chip removal and high feed drive dynamics to realize fast positioning operations [1]. Consequently, moving components should have small mass with sufficient static and dynamic structural stiffness. To allow manufacturers to mill complex parts, five axis milling, where the tool can be set in any direction in space, is state of the art. In a typical axis configuration, a machine tool rotates on the A and B axis and moves across X, Y and Z in a linear direction. Depending on the machining task, various designs and kinematic structures exist to link those five axes. Also, various parallel and hybrid kinematics exist [3].

Among various available drive concepts for machine tool axes, ball screw spindle, rack and pinion, and linear motor drives are typically used in single- or multi-drive arrangements [2]. Depending on different requirements, such as machining process, operating range, machine tool kinematics and desired trajectory dynamics, a best fitting drive concept for each application may be found [7]. The resulting dynamical system is of multiple input - multiple output (MIMO) type, nonlinear and thus pose-dependent.

In industrial applications, the use of cascaded feedback control loops with some extensions such as feedforward controls, filters and reference models is well established

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 120–129, 2024.
https://doi.org/10.1007/978-3-031-50000-8_11

[9]. These independently controlled drives can be tuned without extensive modelling in advance [16]. Automatic tuning algorithms determine control parameters quickly. However, in case of strong pose-dependent axis characteristics, commissioning can become a very challenging task, requiring a lot of expert knowledge. The resulting control loops often suffer from low bandwidth and high trajectory error [2]. Finding robust controller parameters is amongst the most important tasks in machine tool engineering. Although other linear and nonlinear control methods are hard to implement in industrial numerical controllers, they are still of great interest for machine tool manufacturers. Nonlinear position control methods such as flatness-based design [11] have already been used experimentally for parallel kinematic machine tools and show the potential to significantly increase the machine tool performance by reduced tracking error [4].

Mechatronic system simulation offers the possibility to virtually examine machine tool structures, drive concepts, control strategies and machining processes regarding the impact on the overall system behavior. Machine tool dynamics can be assessed and improved in advance, reducing the need for physical test benches [1].

In this paper, the mechatronic system simulation introduced in [7, 13, 14] is extended to multi-drive arrangements like gantry and main-sub, and applied to the coupled machine axis configuration of a linear Y- and rotating A- axis of a recently designed high performance machine tool. To reduce the trajectory error, the standard controllers are extended by a flatness-based feedforward control. Experimental measurements in frequency and time domain of the machine tool with standard controllers show excellent agreement with the mechatronic system simulation. Simulation results demonstrate that nonlinear feed forward control in combination with well-known standard controllers improve positioning performance, e.g. reduced tracking error, overshoot and ringing, significantly.

2 Mechatronic System Simulation

The most comprehensive mechatronic simulation can be reached using flexible multibody systems where large nonlinear feed drive movements can be considered and elastic deformations are included [1]. However, this type of simulation is usually limited to computationally expensive time domain analysis. If movements are assumed to be small, also the finite element method is suitable for a coupled simulation. In this case, the behavior of a linearized system can be even examined in frequency domain. In the present work, a finite element model of an entire machine tool is used to define the mechanical properties of structure and drive trains. Controllers and electrical characteristics are modeled in a graphical block diagramming tool. The approach is similar to an application in references [7, 13, 14] but as novelty also incorporates rotary axes and multi-drive arrangements. Furthermore, a nonlinear rigid multibody model is used to design advanced feed forward control.

2.1 Finite Element Model

Using the finite element method (FEM), the structure of a machine tool including all mechanical parts of the drive trains is approximated by a finite number of regular elements. The remaining unknowns are the displacements of the nodes which are the f

degrees of freedom (DOF) of the system. The equations of motion can be written in matrix form as system of f coupled, second-order, ordinary differential equations (ODEs)

$$\boldsymbol{M}\ddot{\boldsymbol{x}}(t) + \boldsymbol{D}\dot{\boldsymbol{x}}(t) + \boldsymbol{K}\boldsymbol{x}(t) = \boldsymbol{P}(t)\,, \tag{1}$$

where $\boldsymbol{M} \in \mathbb{R}^{f\times f}$, $\boldsymbol{D} \in \mathbb{R}^{f\times f}$, $\boldsymbol{K} \in \mathbb{R}^{f\times f}$ are the global mass, viscous damping, and stiffness matrix, respectively. $\boldsymbol{P} \in \mathbb{R}^{f}$ is the time dependent load vector and $\boldsymbol{x} \in \mathbb{R}^{f}$ is the displacement vector of the degrees of freedom.

For efficiency reasons a normal modes model order reduction of Eq. (1) is performed before rearranging into state-space a representation

$$\begin{bmatrix}\ddot{\boldsymbol{\gamma}}\\ \dot{\boldsymbol{\gamma}}\end{bmatrix} = \underbrace{\begin{bmatrix}-\boldsymbol{m}^{-1}\boldsymbol{d} & -\boldsymbol{m}^{-1}\boldsymbol{k}\\ \boldsymbol{I} & \boldsymbol{0}\end{bmatrix}}_{\boldsymbol{A}} \begin{bmatrix}\dot{\boldsymbol{\gamma}}\\ \boldsymbol{\gamma}\end{bmatrix} + \underbrace{\begin{bmatrix}\boldsymbol{m}^{-1}\boldsymbol{\Phi}^T\boldsymbol{R}^T\\ \boldsymbol{0}\end{bmatrix}}_{\boldsymbol{B}} \boldsymbol{u}\,, \tag{2a}$$

$$\begin{bmatrix}\ddot{\boldsymbol{x}}_a\\ \dot{\boldsymbol{x}}_a\\ \boldsymbol{x}_a\end{bmatrix} = \underbrace{\begin{bmatrix}-\boldsymbol{S\Phi m}^{-1}\boldsymbol{d} & -\boldsymbol{S\Phi m}^{-1}\boldsymbol{k}\\ \boldsymbol{S\Phi} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{S\Phi}\end{bmatrix}}_{\boldsymbol{C}} \begin{bmatrix}\dot{\boldsymbol{\gamma}}\\ \boldsymbol{\gamma}\end{bmatrix} + \underbrace{\begin{bmatrix}\boldsymbol{S\Phi m}^{-1}\boldsymbol{\Phi}^T\boldsymbol{R}^T\\ \boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}}_{\boldsymbol{D}} \boldsymbol{u}\,, \tag{2b}$$

with system matrix $\boldsymbol{A} \in \mathbb{R}^{2m\times 2m}$, input matrix $\boldsymbol{B} \in \mathbb{R}^{2m\times i}$, output matrix $\boldsymbol{C} \in \mathbb{R}^{3j\times 2m}$, feed-through matrix $\boldsymbol{D} \in \mathbb{R}^{3j\times i}$ and modal displacement vector $\boldsymbol{\gamma} \in \mathbb{R}^{m}$. System matrix $\boldsymbol{A}$ holds the modes of the undamped system which are obtained as solution of the generalized eigenvalue problem

$$\boldsymbol{M\Phi\Omega} = \boldsymbol{K\Phi}\,, \tag{3}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{f\times m}$ is the matrix of mass normalized mode shapes, with m representing the number of desired modes, $\boldsymbol{\Omega} = \operatorname{diag}\left(\omega_1^2,\dots,\omega_m^2\right)$ is a diagonal matrix of the eigenvalues ω_k, $k = 1,\dots,m$ and $\boldsymbol{m} = \boldsymbol{\Phi}^T\boldsymbol{M\Phi}$ and $\boldsymbol{k} = \boldsymbol{\Phi}^T\boldsymbol{K\Phi}$ are the modal mass and stiffness matrix, respectively. A modal damping matrix $\boldsymbol{d} = 2\boldsymbol{m}\operatorname{diag}(\xi_1\omega_1,\dots,\xi_m\omega_m)$ is generated by assigning modal damping values ξ_k individually to each mode. Partition matrix $\boldsymbol{S} \in \mathbb{R}^{j\times f}$ reduces the output to j DOFs of interest of displacement vector $\boldsymbol{x}$ in Eq. (1), i.e. the actual nodal displacement $\boldsymbol{x}_a$, velocity $\dot{\boldsymbol{x}}_a$ and acceleration $\ddot{\boldsymbol{x}}_a$ of the measurement systems and some prominent points like tool and workpiece center point (TCP, WCP). Similarly, partition matrix $\boldsymbol{R} \in \mathbb{R}^{i\times f}$ reduces the load vector $\boldsymbol{P}$ of Eq. (1) and limits the possible input vector $\boldsymbol{u} \in \mathbb{R}^{i}$ to forces and moments acting on i outstanding DOFs. To enable a subsequent coupled simulation of a controlled moving mechanical structure, the rigid body modes ($\omega_k = 0$) of the statically underdetermined FEM system have to represent the kinematic permissions of the real machine tool.

2.2 Control Loop Model

In modern machine tools, cascaded control loop structures with innermost PI current control loop, PI velocity control loop and outermost P position control loop are state of the art [9]. Extensions, such as adaptive control parameters, reference models and filters can influence the control loops behavior significantly. The tracking error is reduced using additional feed forward actions. For large scale machine tool axes or carriages with low structural stiffness, drive torque is frequently provided by multiple, mutually independent drives, which improves the overall dynamical properties [16]. In numerical controls, e.g. Siemens Sinumerik NC, typical representatives of these multi-axis couplings are gantry and main-sub [12].

Gantry In a gantry axis grouping a guide axis is traversed in conjunction with at least one synchronized axis. Each drive constitutes a complete axis system and thus has its own measuring system. Setpoint values are provided by the guide axis. To prevent any damage to the machine, all involved gantry drives must be operated in absolute synchronism.

Main-Sub Main-sub coupling is a speed setpoint coupling between a main axis and some sub axes, involving a torque equalization controller for even torque distribution. The sub axis is traversed only speed-controlled, not position-controlled, with the setpoint speed of the main axis. An additional torque can be applied to achieve a mutual tension in the individual drives allowing for the compensation of backlash effects, e.g. gear backlash.

Figure 1 depicts schematically the principal implementation of the control loop of two main-sub coupled axes in the simulation model. Each drive system provides mechanical torque in the motor (M_1, M_2). The actual current i_a is controlled in the innermost PI current control loop by use of proportional gain K_{pi} and integrator reset time T_{ni}. The actual rotational velocity n_a of the motor encoder (E_1, E_2) is regulated towards the setpoint velocity n_s by the velocity PI controller (K_{pn}, T_{nn}). The P position control gain K_v amplifies the tracking error, i.e. the difference of position setpoint x_s and actual virtually measured position x_a. Velocity and current feedforward (n_{ffw}, i_{ffw}) reduce the tracking error considerably but can not affect the disturbance characteristics. Main axis position control provides the velocity setpoint for both, main and sub axis. A PI torque equalization controller (K_{pp}, T_{np}) is involved to distribute the torque evenly to both drives by generating an extra velocity setpoint for each motor. The optional tensioning torque M_p can be applied smoothly by a prefilter with equivalent time T_{ep}.

All feedback loops in the proposed mechatronic system simulation are modelled in discrete-time representation with real-world sampling rates considering delay times for digital measurement acquisition, data transmission and computing times. In addition to the approach presented in previous works [7, 13, 14], this paper also accounts for rotary drives and multi-axis coupling.

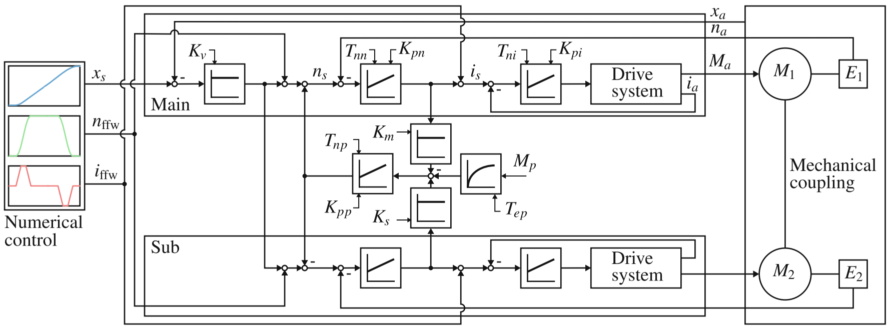


Fig. 1. Schematic representation of the cascaded control loop structure of two main-sub coupled drive axes including torque equalization control and tensioning torque.

2.3 Exact Feedforward and Trajectory Planning

Differential flatness is a nonlinear modelling approach, first introduced by Fliess et al. in 1992 [8]. It extends the notion of controllability from linear systems to nonlinear dynamical systems. In a differential flat system an output exists, with which all states and inputs can be expressed explicitly in terms of this flat output and a finite number of its derivatives.

In the MIMO case, a system of n nonlinear first order differential equations with m affine inputs $\boldsymbol{u} = \begin{bmatrix} u_1 \dots u_m \end{bmatrix}^T \in \mathbb{R}^m$ and m outputs $\boldsymbol{y} = \begin{bmatrix} y_1 \dots y_m \end{bmatrix}^T \in \mathbb{R}^m$ reads,

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \sum_{j=1}^{m} \boldsymbol{g}_j(\boldsymbol{x}) u_j \tag{4a}$$

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}) \tag{4b}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ are the states, $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{g}_j(\boldsymbol{x}),\ j = 1,\dots,m$ are smooth vector fields and $h_l(\boldsymbol{x}),\ l = 1,\dots,m$ are smooth functions.

Derivatives of all $h_l(\boldsymbol{x})$ outputs can be noted elegantly using Lie-derivatives [15] and yield the transformation

$$\boldsymbol{z} = \boldsymbol{\Phi}(\boldsymbol{x}) = \left[h_1(\boldsymbol{x}),\, L_{\boldsymbol{f}} h_1(\boldsymbol{x}),\, \dots,\, L_{\boldsymbol{f}}^{(r_1-1)} h_1(\boldsymbol{x}),\, h_2(\boldsymbol{x}),\, \dots,\, L_{\boldsymbol{f}}^{(r_m-1)} h_m(\boldsymbol{x}) \right]^T , \tag{5}$$

and consequently

$$\begin{bmatrix} y_1^{(r_1)} \\ \vdots \\ y_m^{(r_m)} \end{bmatrix} = \underbrace{\begin{bmatrix} L_{\boldsymbol{f}}^{(r_1)} h_1(\boldsymbol{x}) \\ \vdots \\ L_{\boldsymbol{f}}^{(r_m)} h_m(\boldsymbol{x}) \end{bmatrix}}_{\boldsymbol{b}(\boldsymbol{x})} + \underbrace{\begin{bmatrix} L_{\boldsymbol{g}_1} L_{\boldsymbol{f}}^{(r_1-1)} h_1(\boldsymbol{x}) & \dots & L_{\boldsymbol{g}_m} L_{\boldsymbol{f}}^{(r_1-1)} h_1(\boldsymbol{x}) \\ \vdots & \ddots & \vdots \\ L_{\boldsymbol{g}_1} L_{\boldsymbol{f}}^{(r_m-1)} h_m(\boldsymbol{x}) & \dots & L_{\boldsymbol{g}_m} L_{\boldsymbol{f}}^{(r_m-1)} h_m(\boldsymbol{x}) \end{bmatrix}}_{\boldsymbol{D}(\boldsymbol{x})} \begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix} . \tag{6}$$

If $\boldsymbol{D}(\boldsymbol{x})$ is regular in a vicinity $\bar{\boldsymbol{x}} \in \mathscr{U}$, the vector relative degree $r = r_1 + \dots + r_m = n$ and the desired trajectory matches the systems initial conditions $\boldsymbol{x}_0 = \boldsymbol{x}(0) = \boldsymbol{x}_d(0)$, the system can be exactly linearized and feed forward u_j^* can be planned using

$$\begin{bmatrix} u_1^* \\ \vdots \\ u_m^* \end{bmatrix} = \boldsymbol{D}^{-1}(\boldsymbol{x}_d) \left(\begin{bmatrix} y_{1,d}^{(r_1)} \\ \vdots \\ y_{m,d}^{(r_m)} \end{bmatrix} - \begin{bmatrix} L_{\boldsymbol{f}}^{(r_1)} h_1(\boldsymbol{x}_d) \\ \vdots \\ L_{\boldsymbol{f}}^{(r_m)} h_m(\boldsymbol{x}_d) \end{bmatrix} \right), \tag{7}$$

where $\boldsymbol{x}_d = \boldsymbol{\Phi}^{-1}\left(\left[y_{1,d}\ \dot{y}_{1,d}\ \ldots\ y_{1,d}^{(r_1-1)}\ y_{2,d}\ \ldots\ y_{m,d}^{(r_m-1)}\right]^T\right)$ is computed from the desired trajectory of the outputs and their derivatives. Consequently, the planned trajectories $y_{l,d}$, $l = 1,\ldots,m$ must be at least r_l-times continuously differentiable to result in a continuous system input. However, if the chosen outputs result in $r = r_1 + \ldots + r_m < n$, an internal zero dynamics remains in the system.

Among many existing approaches for trajectory planning, polynomial functions [10], Gevrey functions [5] and classical s-curve trajectories whose differentiability is increased through filters [6] are common. In this work polynomial functions for rest-to-rest trajectories of form

$$y_{l,d}(t) = y_{l,d,S} + \left(y_{l,d,E} - y_{l,d,S}\right)\left(\sum_{i=r_l+1}^{2r_l+1} \tilde{\gamma}_i \left(\frac{t-t_S}{t_E-t_S}\right)^i\right) \tag{8}$$

are used, where $y_{l,d,S} = y_{l,d}(t_S)$ is the start position and $y_{l,d,E} = y_{l,d}(t_E)$ is the end position, respectively. The requirement that all derivatives must be zero at the start and end point yields the coefficients $\tilde{\gamma}_i$ of the polynomial [10]. The transition time $t_T = t_E - t_S$ of the rest-to-rest movement can be optimized offline in a way, that all requirements regarding maximum acceleration, velocity and drive torque of the involved axes are met.

3 Application, Results and Discussion

In this work, the recently designed standard machine center *syncromill c21-63/1500* of machine tool manufacturer Fill GmbH is considered. In particular its Y- and A- axis are of interest. Figure 2 depicts an overview of the machine tools axes configuration with the associated drives. Workpieces are mounted to a carriage (bridge) which traverses in the vertical Y- direction and rotates around the A- axis. The two axes form an open kinematic chain whose system dynamics can be formulated as nonlinear MIMO system. A mechatronic system simulation as proposed in Sect. 2 has been set up to investigate the individual axis dynamics. Simulations revealed that for the vertical movement two drives (Y1 and Y2) with gantry grouping are necessary to avoid unacceptable canting of the bridge. For the A-axis rotation, both sides of the bridge are driven in a gantry group. Each individual side consists of two motors in main-sub coupling engaging into a common gear wheel with mutual tensioning torque.

The machine tool is equipped with a Siemens Sinumeric One CNC system, which offers a built-in data logger to acquire measurement data in time and frequency domain. Figure 3 (a) depicts the measured x_{meas} and simulated x_{sim}, $x = (A,Y)$ open loop frequency response function (FRF) of the velocity controlled axes. These FRFs describe

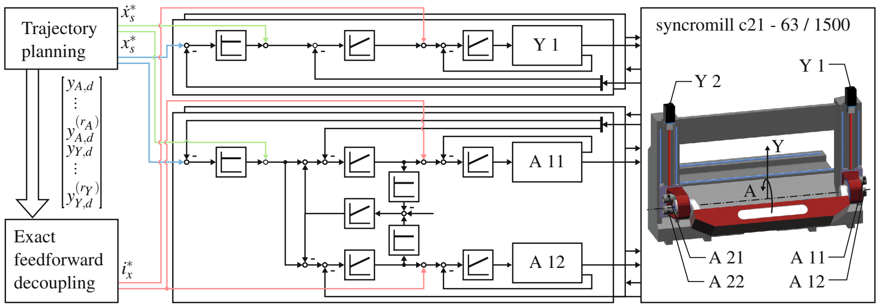


Fig. 2. Schematic representation of the nonlinear MIMO feed forward control in simulation studies. Frames offset in the background represent identical accompanying gantry group control loops.

in frequency domain the actual motor velocity output at given actual torque input and are considered as plant in control theory. Furthermore, in (b) and (c) measured and simulated data of closed loop velocity and position control is given. Note that simulated FRF data results from the comprehensive mechatronic system model, linearized at the specified machine tool pose. Considering the enormous complexity of the application, all measured and simulated FRFs agree excellently. Figures 5 (a) and (b) show that joint movement of Y and A by standard jerk limited s-curve trajectories, e.g. the acceleration characteristic is trapezoidal, excites the individual position control resonance frequency significantly. This results in oscillations approaching the end position. Furthermore, individual single axis operation (c) and (d) leads to strong disturbance of the respective non-moving axis.

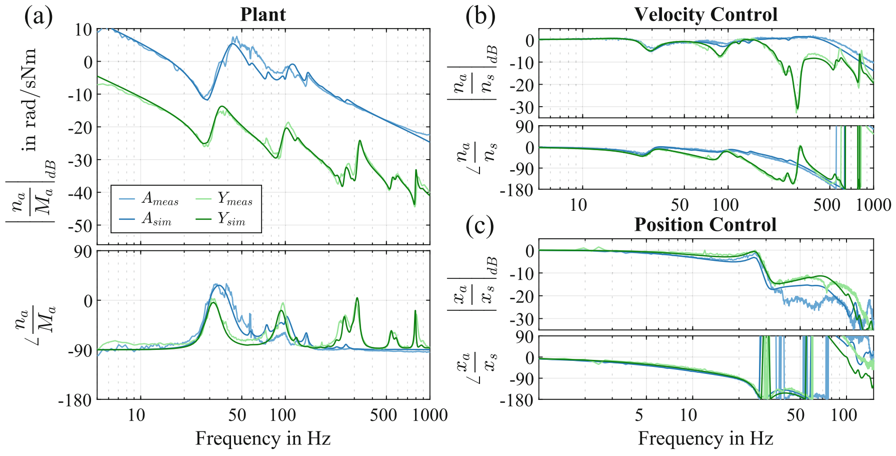


Fig. 3. Measured and simulated FRFs at position $Y = 0$, $A = 90°$: Velocity controlled system (a); Closed loop velocity control (b); Closed loop position control (c)

To improve the positioning performance, exact feed forward control based on trajectory planning of differential flat outputs has been developed. For exact feed forward planning a model covering the main nonlinearities of the Y- and A- axis combination is necessary. Figure 4 provides a sketch of the lumped model of this nonlinear system. Inertia J_{MY} accounts for the motors, couplings and spindles of the Y- axis. The motor rotation ϕ_Y is transformed into a vertical movement y via pitch p by a ball screw spindle, modelled as spring c_N and damper d_N. No posture dependency of c_N is considered, since the associated natural frequency occurs in the middle frequency range, which is of little interest for the position accuracy. A rigid guiding system connects the Y- carriage (m_Y) to the substructure. Load and bridge (m_L, J_L) are pivoted in the Y- carriage and rotate with ϕ_L. The bridge is connected to a gearbox i_G and subsequently via spring c_G and damper d_G to the motors of the A- axis (J_{MA}). Input torque M_A and M_Y is provided by motors of A and Y which rotate with ϕ_A and ϕ_Y, respectively. Although the overall machine tool installation condition and the guiding system influence the oscillations during positioning operations significantly, it is neglected as its consideration by additional degrees of freedom would detract the flatness property from the selected system outputs ϕ_L and y. The four generalized variables ($f = 4$) of the lumped system are $\boldsymbol{q} = \left[\phi_Y \; y \; \phi_A \; \phi_L\right]^T$. Lagrange's equation yield equations of motion

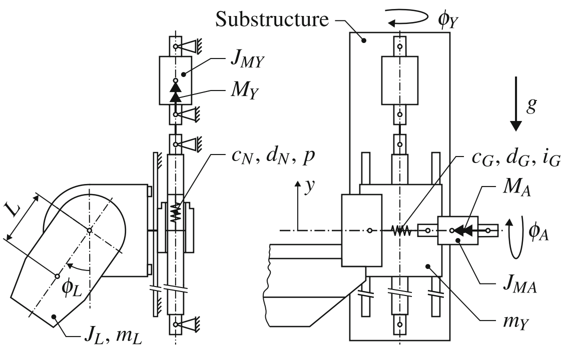


Fig. 4. Nonlinear, lumped, rigid multibody model of machine tool A- and Y- axis.

$$\frac{d}{dt}\left[\frac{\partial\left(E_{kin}-E_{pot}\right)}{\partial \dot{q}_k}\right]-\frac{\partial\left(E_{kin}-E_{pot}\right)}{\partial q_k}+\frac{\partial R}{\partial \dot{q}_k}=Q_k\,, \qquad k=1,\ldots,f \tag{9}$$

where E_{kin} holds all the systems kinetic energy and E_{pot} the potential energy, respectively. Rayleigh dissipation function R handles the effects of the velocity-proportional frictional forces of the outputs and on the right hand side Q_k considers all non-conservative forces and moments like M_A and M_Y. Introduction of state vector $\boldsymbol{x} = \left[\phi_Y \; \dot{\phi}_Y \; y \; \dot{y} \; \phi_A \; \dot{\phi}_A \; \phi_L \; \dot{\phi}_L\right]^T$, yields a system of nonlinear, first order differential equations with two affine inputs,

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}_Y(\boldsymbol{x})M_Y + \boldsymbol{g}_A(\boldsymbol{x})M_A \ , \tag{10a}$$

$$y = h_Y(\boldsymbol{x}) \ , \tag{10b}$$

$$\phi_L = h_A(\boldsymbol{x}) \ . \tag{10c}$$

Due to the lengthy expressions, this system (10) is not given explicitly. For the described model the selected outputs ϕ_L and y are flat, inputs M_A and M_Y enters explicitly in derivatives $r_Y = r_A = 4$. Figure 2 gives an overview of the feed forward control in simulation studies. The individual cascaded control loops including main-sub and gantry groupings remain unaltered due to partly inaccessible interfaces and capsuled Siemens Sinamics S120 drive units. The drives receive setpoint and feedforward values for position $x_s^* = y_{x,d}$, velocity $\dot{x}_s^* = \dot{y}_{x,d}$ and motor current i_x^*, $x = (A, Y)$ generated from exact feedforward decoupling (7) of desired polynomial trajectories (8).

For fair comparison purpose in Fig. 5, the transition time t_T of the proposed polynomial setpoint trajectories x_s^* and standard jerk-limited s-curve setpoint trajectories x_s is equal. Due to the higher smoothness of the polynomial trajectory and the almost perfectly feed forwarded drive torque, less oscillations are excited in simulated actual position signals $x_{a,sim}^*$ compared to measured and simulated actual position signals $x_{a,meas}$ and $x_{a,sim}$. The disturbance caused by the respective other moving axis is also reduced considerably. However, non-equal time constants of the velocity and current controllers as well as transmission delays prevent even better disturbance characteristics.

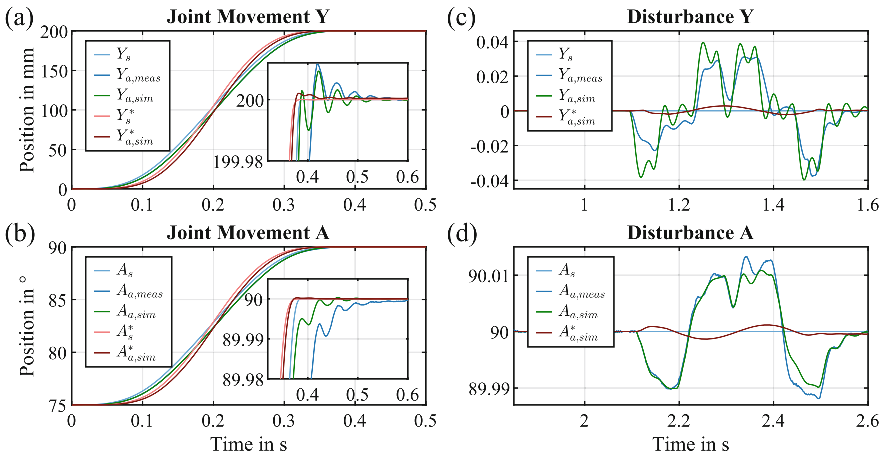


Fig. 5. Measured and simulated rest-to-test movements of standard s-curve trajectories with standard feed forward, compared to simulated polynomial trajectories with proposed exact feed forward: Joint positioning operation of Y axis (a) and A axis (b); Y axis response during A axis moving (c); A axis response during Y axis moving (d)

4 Conclusion

The developed mechatronic system simulation can implement all common machine tool drive systems including multidrive applications such as main-sub or gantry. It is an excellent analysis tool to evaluate the mechanical structure, axis kinematics, drive systems and control. The authors are confident that for Siemens Sinumeric One a nonlinear MIMO exact feed forward control can be implemented using so called compile cycles. Simulation studies promise a significant improvement of positioning performance using this exact feed forward control.

References

1. Altintas, Y., Brecher, C., Weck, M., Witt, S.: Virtual machine tool. CIRP Ann. **54**(2), 115–138 (2005)
2. Altintas, Y., Verl, A., Brecher, C., Uriarte, L., Pritschow, G.: Machine tool feed drives. CIRP Ann. **60**(2), 779–796 (2011)
3. Altuzarra, O., Hernández, A., San Martín, Y., Larrañaga, J.: Parallel kinematics for machine tools. In: López de Lacalle, L., Lamikiz, A. (eds) Machine Tools for High Performance Machining. Springer, London (2009)
4. Ast, A., Eberhard, P.: Flatness-based control of parallel kinematics using multibody systems - simulation and experimental results. Arch. Appl. Mech. **76**, 181–197 (2006)
5. Bachmayer, M., Ulbrich, H., Rudolph, J.: Flatness-based control of a horizontally moving erected beam with a point mass. Math. Comput. Model. Dyn. Syst. **17**(1), 49–69 (2011)
6. Beckmann, D., Schappler, M., Dagen, M., Ortmaier, T.: New approach using flatness-based control in high speed positioning: experimental results. IEEE Int. Conf. Ind. Tech., 351–356 (2015)
7. Binder, R., Wiesauer, M.: Analyzing the impact of different drive concepts on machine tool dynamics using mechatronic system simulation. In: Proceedings of NAFEMS World Congress (2021)
8. Fliess, M., Lévine, J., Martin, P., Rouchon, P.: On differentially flat nonlinear systems. IFAC Proc. **25**(13), 159–163 (1992)
9. Groß, H., Hamann, J., Wiegärtner, G.: Elektrische Vorschubantriebe in der Automatisierungstechnik. Publicis Corporate Publishing (2006)
10. Lévine, J.: Analysis and control of nonlinear systems. A flatness-based approach. Springer, Berlin, Heidelberg (2009)
11. Rothfuß, R.: Anwendung flachheitsbasierter Analyse und Regelung nichtlinearer Mehrgrößensysteme. Düsseldorf: Fortschrittsberichte VDI, Reihe 8: Meß- Steuerungs- und Regelungstechnik, Nr. 664, VDI Verlag (1997)
12. Siemens, A.G.: Sinumerik one, axes and spindles, function manual. Siemens AG Digital Industries (2022)
13. Wiesauer, M., Bleicher, F.: Parameterization method for non-linear friction models of machine tool feed drives. Proc. CIRP **102**, 399–404 (2021)
14. Wiesauer, M., Habersohn, C., Bleicher, F.: Validation of a coupled simulation for machine tool dynamics using a linear drive actuator. J. Manuf. Mater. Process. **5**, 1 (2021)
15. Yano, K.; Lévine, J.; Martin, P.; Rouchon, P.: The Theory of Lie Derivatives and its Applications. North-Holland (1957)
16. Zirn, O.: Machine tool analysis. modelling, simulation and control of machine tool manipulators. ETH Zürich (2008)

Feedforward Control for a Manipulator with Flexure Joints Using a Lagrangian Neural Network

Eline Heerze, Bojana Rosic, and Ronald Aarts(✉)

Applied Mechanics and Data Analysis, University of Twente, Enschede, The Netherlands
elineheerze@hotmail.com, {B.Rosic,R.G.K.M.Aarts}@utwente.nl

Abstract. Feedforward control of a manipulator can be generated with a sufficiently accurate stable inverse model of the manipulator. A Feedforward Neural Network (FNN) can be trained with experimental data to generate feedforward control without knowledge about the system at hand. However, the FNN output can show unphysical behaviour especially in operational regimes where the training data is sparse. Instead, the output of a Lagrangian Neural Network (LNN) is limited by physical constraints and hence is expected to predict the (inverse) multibody system behaviour more robustly. We propose to generate the feedforward control by first training a LNN that captures already most features in experimental data and next add a FFN to account for a relatively small residual. Experimental results from a fully actuated 2-DOF manipulator with flexure joints show that the accuracy of the controlled motion using this approach is comparable to using an identified inverse plant model built from the system's equations of motion.

1 Introduction

Feedforward control can greatly improve the accuracy of a manipulator. In a typical implementation an inverse dynamic model of the multibody system at hand is used to compute the required actuator input to follow the reference trajectory. The achievable performance gained from this feedforward control depends heavily on the correctness and completeness of the model. In a *model-based* approach a *white-box* model with the equations of motion of the multibody system is derived and its parameters are estimated [6,10]. Assuming these parameters have a clear physical meaning, it is expected that the model can be used for a wide variety of trajectories. However, the "richness" of the model is obviously restricted to the features included in the model structure.

Alternatively, in a *data-driven* approach a *black-box* model is identified purely from data e.g. using machine learning techniques. Abdul-hadi [1] presents a Feedforward Neural Network (FNN) to learn the dynamic behaviour of a robotic system without any knowledge about the system dynamics and its parameters. It proved to be feasible to solve this problem with reasonable accuracy. However, care has to be taken to avoid overfitting and likely the model outputs are incorrect for operating conditions that were not sufficiently excited in the training data.

To mitigate this risk physics informed neural networks (PINN) are being researched, where the training is constrained to a predefined physical law. Lutter *et al.* [8,9] propose a Lagrangian Neural Network (LNN), or Deep Lagrangian Networks (DeLaN), to incorporate the Lagrangian dynamics into a neural network. The authors could obtain an inverse dynamic model of a robotic system of which the accuracy is similar to the performance of an analytical model. Furthermore, the trained model can handle extrapolations to new trajectories, e.g. with increased velocities. One drawback of this LNN is the exclusion of non-conservative forces like damping and friction in the Lagrangian formulation which can result in significant errors when applied to real physical systems. Liu *et al.* [7] extend an LNN with a parallel FNN to approximate non-conservative physics. Both neural networks are trained simultaneously, where the unconstrained FNN is penalised to discourage it from learning conservative dynamics. However, it was found that the results strongly depend on the penalisation factor.

In this paper we research the use of combining a LNN and FNN to learn an inverse dynamic model of a manipulator with flexure joints [3]. For such manipulator it is expected that the conservative contribution dominates. Hence it is proposed to train the neural networks sequentially: First the LNN should capture the dominant conservative dynamic behaviour and next the FNN is trained with the relatively small residue. The outputs of both networks are added to estimate the actuator forces needed to perform a specified motion. This estimation is applied as feedforward control and the improvement of the motion accuracy is compared to results obtained with feedforward control computed with a white-box manipulator model.

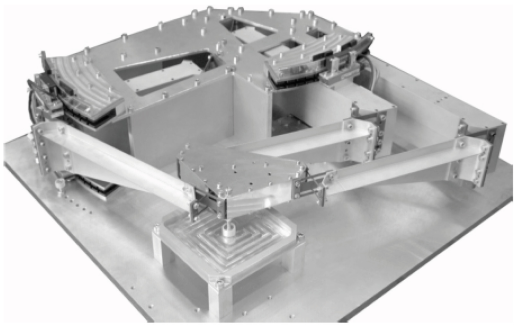
2 Method

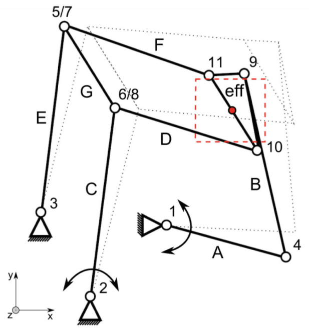
2.1 2-DOF Manipulator with Flexure Joints

We consider the fully actuated manipulator with two degrees of freedom (DOF) shown in Fig. 1(a) [3]. The schematic drawing of Fig. 1(b) illustrates that two actuators drive the rotation of two arms ("A" and "C") resulting in a translational end-effector ("eff") motion in a horizontal plane. All joints are flexure joints allowing rather smooth operation with low friction and hysteresis. Consequently, contributions from the link mass and joint stiffness dominate in the non-linear equation of motion written in independent generalised coordinates $\boldsymbol{q}$ as

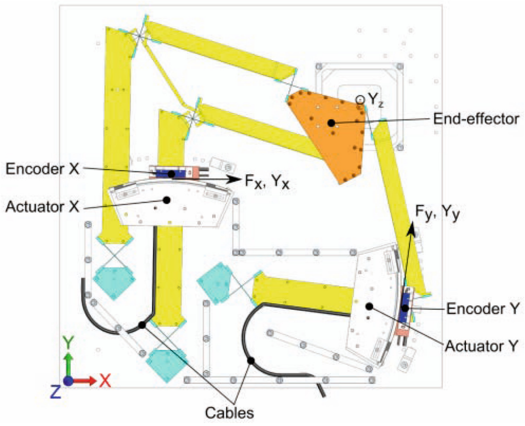
$$\boldsymbol{F} = \boldsymbol{F}_{\mathrm{c}} + \boldsymbol{F}_{\mathrm{nc}} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \frac{1}{2}\dot{\boldsymbol{q}}^T \frac{\partial \boldsymbol{M}(\boldsymbol{q})}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial V(\boldsymbol{q})}{\partial \boldsymbol{q}} + \boldsymbol{F}_{\mathrm{nc}}, \tag{1}$$

where $\boldsymbol{F}_{\mathrm{c}}$ represents the conservative part in the total actuator force vector $\boldsymbol{F}$. It is expressed in the symmetric and positive definite mass matrix $\boldsymbol{M}$ and potential energy V, the latter due to the stiffness in all joints. The non-conservative


(a) Photo (by Ger Folkersma).



(b) Schematic view.



(c) Top view.

Fig. 1. The 2 DOF manipulator with flexure joints [3]. The view of the manipulator in the photo is rotated relative to the other views.

forces $\boldsymbol{F}_{\text{nc}}$ describe all remaining, possibly non-linear effects like cogging or friction caused by the cables connected to the moving parts of the actuators and sensors. Both rotation angles of the actuated arms are chosen as independent coordinates $\boldsymbol{q}$. These are computed from the displacements Y_x and Y_y measured with "Encoder X" and "Encoder Y" in Fig. 1(c). Both actuator forces F_x and F_y shown in this figure are calculated from the applied motor currents assuming a constant and known ratio between force and current for each motor.

2.2 White-Box Model

For the white-box model the kinematic relations must be derived that express the motion of all links and the joint rotations in the independent coordinates $\boldsymbol{q}$. The kinematic expressions are simplified by ignoring pivot shifts that are inherently introduced by the cross flexure joints used for the joints in this setup.

In the dynamic parameters 5 independent mass and inertia contributions from the links and end-effector can be defined. To represent the joint stiffnesses

6 independent stiffness parameters are needed. The damping and friction contribution $\boldsymbol{F}_{\mathrm{nc}}$ is assumed to be dominated by 2 damping parameters that capture the (viscous) damping arising from the cables of which the deformations are directly linked to the joint rotations and hence the independent coordinates $\boldsymbol{q}$. Finally it was observed in experimental data that the measured forces can exhibit a constant offset which results in 2 more parameters.

The equation of motion (1) can be written in a parameter linear expression as

$$\boldsymbol{\Phi}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\ \boldsymbol{\theta} = \boldsymbol{F}, \tag{2}$$

where parameter vector $\boldsymbol{\theta}$ stores the 15 parameters and regression matrix $\boldsymbol{\Phi}$ depends only on $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$. From a mathematical point of view it is expected that the regression matrix is non-singular for the proposed parameter set and all parameters can be found with the Moore–Penrose pseudoinverse $\boldsymbol{\Phi}^{\dagger}$. However, using experimental data it may appear that with a feasible excitation not all parameters can be identified accurately. Common linear regression techniques can be used to determine a base parameter vector, e.g. from a singular value decomposition of $\boldsymbol{\Phi}$ [6,10].

2.3 Lagrangian Neural Network (LNN)

To obtain the black-box model it would be possible to learn the total forces $\boldsymbol{F}$ from trajectory data $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ with a FNN, but then the physical structure of $\boldsymbol{F}_{\mathrm{c}}$ is not taken into account. Hence a LNN [8,9] is used for this part of the feedforward control that can be written as

$$\boldsymbol{F}_{\mathrm{c}} = \boldsymbol{L}(\boldsymbol{q})\boldsymbol{L}^{T}(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + \frac{1}{2}\dot{\boldsymbol{q}}^{T}\frac{\partial(\boldsymbol{L}(\boldsymbol{q})\boldsymbol{L}^{T}(\boldsymbol{q}))}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial V(\boldsymbol{q})}{\partial \boldsymbol{q}}, \tag{3}$$

where the lower triangular matrix $\boldsymbol{L}$ is the Cholesky decomposition of the mass matrix $\boldsymbol{M}$. For a 2-DOF system it has only one non-zero off-diagonal term l_{o1}

$$\boldsymbol{L} = \begin{bmatrix} l_{d1} & 0 \\ l_{o1} & l_{d2} \end{bmatrix}, \tag{4}$$

and both diagonal terms l_{d1} and l_{d2} are positive to ensure a positive definite mass matrix $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^{T}$. The rightmost term in Eq. (3) includes the non-negative potential energy V.

Figure 2 shows the structure of the LNN implementation in which a "Physics layer" is added to a FNN. This combination and the implementation of the FNN assure that the LNN represents Eq. (3). The outputs of the FNN are estimates of the potential energy $\hat{V}$ and matrix $\hat{\boldsymbol{L}}$ where the latter is split into the off-diagonal $\hat{\boldsymbol{l}}_{o}$ and positive diagonal $\hat{\boldsymbol{l}}_{d}$ terms. All estimates are functions of the positions $\boldsymbol{q}$ only. The blue neuron ($\hat{\boldsymbol{l}}_{o}$) in the output layer represents a linear activation function and the orange neurons ($\hat{\boldsymbol{l}}_{d}$, $\hat{V}$) have a ReLu or rectifier activation, which means that their output equals the neuron's input for non-negative values and zero otherwise. Automatic differentiation [2] is used to compute the derivatives

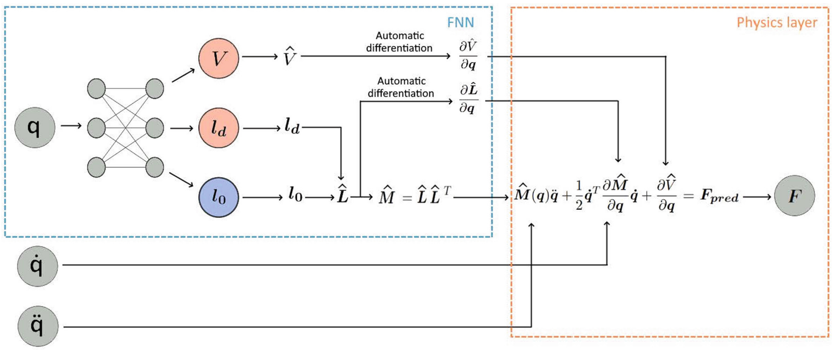

FNN
Physics layer
Automatic differentiation
Automatic differentiation
q
V
l_d
l_0
$\hat{V}$
l_d
l_0
$\hat{L}$
$\frac{\partial \hat{V}}{\partial q}$
$\frac{\partial \hat{L}}{\partial q}$
$\hat{M} = \hat{L}\hat{L}^T$
$\hat{M}(q)\ddot{q} + \frac{1}{2}\dot{q}^T \frac{\partial \hat{M}}{\partial q}\dot{q} + \frac{\partial \hat{V}}{\partial q} = F_{pred}$
F
q̇
q̈

Fig. 2. Structure of the LNN, including the physics transformations (adapted from [5]).

of $\hat{V}$ and $\hat{\boldsymbol{L}}$ with respect to $\boldsymbol{q}$ that appear in Eq. (3). The network is trained to minimise the mean squared error between the estimated forces denoted $\boldsymbol{F}_{pred}$ in Fig. 2 and measured forces.

The remaining forces $\boldsymbol{F}_{\mathrm{nc}}$ are estimated with a general FNN, where it is assumed the forces only depend on positions $\boldsymbol{q}$ and velocities $\dot{\boldsymbol{q}}$. In the training procedure of both networks we prefer to avoid the dependency on a penalisation factor that is used in simultaneous training [7]. We concluded from simulations that training the LNN first on the full $\boldsymbol{F}$ results in a sufficiently accurate estimate of $\boldsymbol{F}_{\mathrm{c}}$. Apparently the relatively small contribution of $\boldsymbol{F}_{\mathrm{nc}}$ does not result in a significantly biased estimate of the $\boldsymbol{F}_{\mathrm{c}}$ part. Hence the LNN is trained first and next the FNN is trained on the residue $\boldsymbol{F} - \boldsymbol{F}_{\mathrm{c}}$. Tensorflow is used for the training of both LNN and FNN.

3 Results

3.1 Training Data

To identify the parameters in the white-box model and to train the neural networks, 9 datasets have been used from as many controlled motion experiments. For each experiment a two-dimensional trajectory has been generated which specifies the desired end-effector x and y positions as functions of time. Each of these desired positions is described with a non-periodic smooth random function [4] defined by Fourier series with random coefficients. For a given wavelength parameter $\lambda > 0$ and interval $[0, L]$, the function is given by

$$f(t) = \sqrt{2} \sum_{j=1}^{m} \left[a_j \cos\left(\frac{2\pi j t}{L}\right) + b_j \sin\left(\frac{2\pi j t}{L}\right)\right], \tag{5}$$

where $m = L/\lambda$ and coefficients a_j, b_j are independent samples from a normal distribution with zero mean and variance $\frac{1}{2m+1}$. The datasets are generated with $\lambda \in [10, 8.0, 6.0, 4.0, 0.9, 0.8, 0.6, 0.5, 0.4]$, a total duration of $L = 120$ s and sample time $t_s = 0.0001$ s. All paths are scaled to a maximum amplitude of ± 30 mm and converted to actuator displacements using inverse kinematic relations assuming rigid links and ideal rotational joints. These paths are the reference positions for controlled actuator motion where a PD feedback controller is used that should result in reasonably accurate tracking. The actual encoder reading and applied motor currents are recorded. The independent coordinates $\boldsymbol{q}$ and actuator forces $\boldsymbol{F}$ are computed as described in Sect. 2.1. Velocities $\dot{\boldsymbol{q}}$ and accelerations $\ddot{\boldsymbol{q}}$ are derived off-line with numerical differentiation of the position data and low-pass filtering.

The collected data in each dataset originally represents 1,200,000 time samples. The first 200,000 samples are discarded to eliminate any initial transient response. The last 200,000 samples are discarded as well, such that 800,000 samples remain in each dataset. Datasets with larger values of λ have more low frequent content and excite in particular the stiffness properties, i.e. the potential energy $V(\boldsymbol{q})$, whereas datasets with smaller λ reveal more mass dominated dynamics. Contributions from the non-conservative forces can appear at various frequencies, depending on the cause of these forces. E.g. viscous damping reduces the significance of the resonance peaks which appear near 1 Hz and 2 Hz. Hence this contribution can be identified from data sets with smaller λ that include these resonance frequencies.

3.2 White-Box Identification

To identify the 13 physical parameters in the white-box model it is not needed to use the large amount of 7,200,000 samples from all 9 datasets. Only 1/1000 of these time steps are used in the following identification. Regarding the offsets mentioned in Sect. 2.2 it was found that datasets 8 and 9 showed a slightly different offset compared to datasets 1–7. Hence 2×2 offset parameters are included in the parameter vector giving rise to 17 parameters in total. However, it was found from a singular value decomposition of the regression matrix $\boldsymbol{\Phi}$ that no more than 12 independent parameters can be identified. This is confirmed from an analysis of the residual error which hardly decreases when more than 12 (linear combinations of) parameters are identified.

Figure 3 shows the measured forces F and the misfit of the forces ΔF for all 9 datasets. The datasets are concatenated in the plot, but as explained in Sect. 3.1 these datasets are collected independently and hence discontinuities can be seen in the plot. The force misfit is less than about 10% compared to the actual force. This error is somewhat larger than expected beforehand, which could be caused by experimental imperfections e.g. from the cables (see Fig. 1(c)) or by (white-box) model errors e.g. due to an incorrect kinematic model in which pivot shift is neglected.

3.3 LNN Training

Initially only the LNN is trained for which also only a relatively small subset of all data is used with 20,000 samples, i.e. every 400th sample. These samples are shuffled after which 70% is used for training and 30% for validation. The

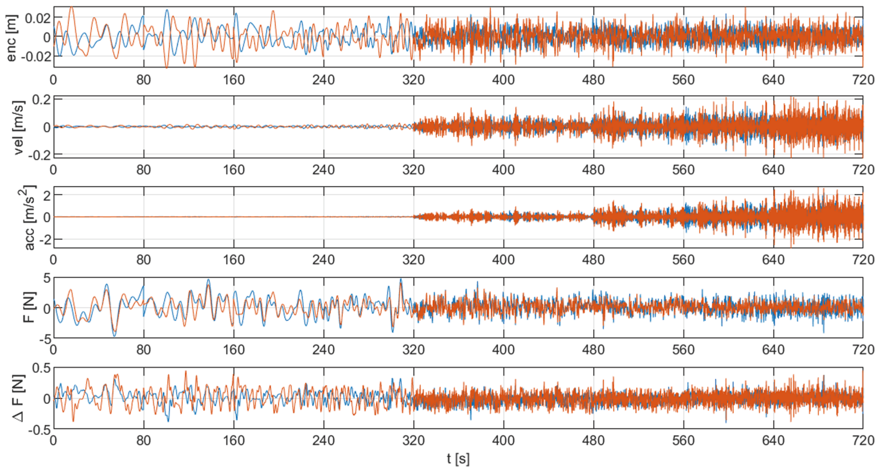


Fig. 3. Manipulator positions, velocities, accelerations, forces F and force misfit ΔF from an identification with 12 independent parameters. The different colours represent x and y data respectively.

Adam solver is used and the batch size is 32. The training is run for 40 epochs with a learning rate $l_r = 0.001$, next 10 epochs with $l_r = 0.0001$ and finally 10 epochs with $l_r = 0.00001$. Furthermore training is initialised 10 times with random seeds. The performance is measured with the Mean Absolute Error (MAE) between target and approximation. The MAE is evaluated for the best training result from all random seeds, as well as for its standard deviation to measure the robustness of the training.

To select the activation function, a network with 6 hidden layers with 32 neurons each is used. The activation function of the output layer is fixed as outlined in Sect. 2.3. Softplus, sigmoid, ReLu and tanh activation functions are evaluated [5]. The latter two give similar results in terms of best MAE, where tanh shows the smallest standard deviation and therefore is the preferred activation function.

For this activation function the effect of varying network size is evaluated next for different numbers of hidden layers and neurons per layer [5]. A network with 32 neurons per layer gives better result where a minimal MAE is found for 8 hidden layers. Hence 8 hidden layers with 32 neurons will be used for the LNN.

For this network two additional check have been done. The number of time samples has been varied. It appeared that the MAE increases when less than 14,000 samples were used, but hardly changes when using 14,000 or 35,000 samples. The trainings presented in this section are well within the latter range. Finally, the batch size and optimizer have been evaluated. It was found that training with a batch sizes of 16, 32 and 64 does not yield a significant change in outcome. Similarly, using Adagrad or SGD instead of the Adam optimiser didn't give significantly different results either.

3.4 FNN Training

Once the LNN is trained as outlined in the previous section, it is fixed and the FNN is trained on the residue. A similar procedure is followed to evaluate the FNN performance and optimise its hyperparameters. Once more the Adam solver is used with a batch size of 32. As no physical structure constrains the FNN, it is expected more training data is needed and more epochs are required before the solution is converged. Therefore the FNN is trained for 100 epochs with $l_r = 0.001$ and next for 50 epochs with $l_r = 0.0001$, initially on a dataset with 50,000 time samples. The input data is scaled to values between -1 and $+1$ using the MinMaxScaler. It appeared that the probability of converging to a local minimum is less likely and hence the training is initialised with only 3 random seeds.

Different activation functions are evaluated for a FNN with 3 hidden layers and 16 neurons in each layer [5]. The tanh activation function is preferred as it results in a small MAE with a small standard deviation.

Next different numbers of hidden layers and neurons per layer are evaluated [5]. It was found that relatively large networks offer a better MAE for the training and validation data. However, when applying the results to an new test dataset it appeared that the larger networks clearly suffer from overfitting. A smaller MAE was obtained with a smaller network. The most suitable FNN structure has only 2 hidden layers with 8 neurons each.

This FNN has been trained with varying numbers of time samples. The MAE is shown in Fig. 4. Clearly a large number of samples is advantageous. Extrapolating the trend in the plots, the number of samples could even be increased beyond the maximum presented in the figure. However, this will also result in an increase of computational costs, so 200, 000 samples will be used to train the FNN.

Table 1 summarises the MAE obtained for training and validation using only the preferred LNN and combining it with the FNN as proposed above. The validation MAE is split to show the errors for both actuator forces F_x and F_y separately. The table illustrates the improvement that can be obtained by adding the FNN to the LNN. It also shows that the errors in both forces are comparable

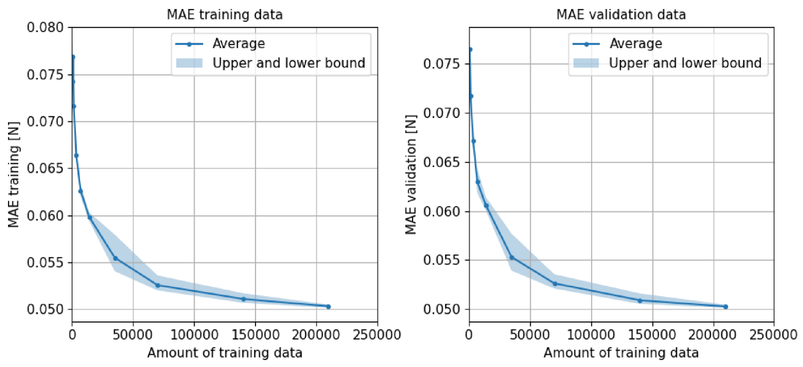


Fig. 4. Mean Absolute Error (MAE) for different numbers of time samples used for the FNN training (2 hidden layers with 8 neurons each) [5].

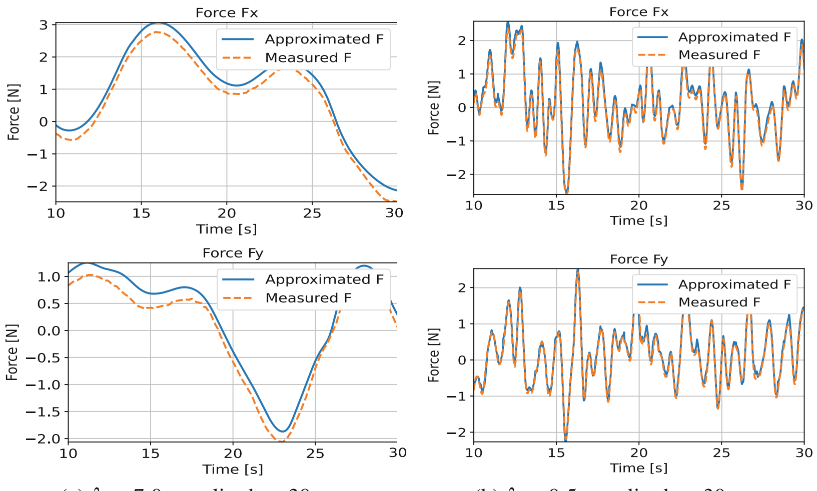
Table 1. MAE on the training and validation data using the selected LNN and FNN. The MAE of the white-box model, Sect. 3.2, is included for comparison.

	Training MAE [N]	Validation MAE [N]	Valid. MAE F_x [N]	Valid. MAE F_y [N]
LNN	0.1022	0.1022	0.0836	0.1208
LNN + FNN	0.0517	0.0514	0.0562	0.0466
white-box model	0.0864			

and are below 2%. Furthermore it appears that the MAE of the force estimates from the combined neural networks is smaller than is obtained with the white-box model of Sect. 3.2.

3.5 Offline Force Estimates

The performance of the white-box model and the trained LNN+FNN combination is evaluated using 10 test trajectories that are generated similarly to the datasets defined in Eq. (5) in Sect. 3.1 except for a shorter duration with $L = 40$ s. The first and last 10 s of the data are removed and for the remaining time samples the forces are estimated with both methods and compared to measured values. Data processing is similar to the procedure outlined in Sect. 3.1 for the training data, i.e. the trajectories are the reference positions for controlled actuator motion. The forces are estimated "offline" which means that the motion data have been captured to be processed afterwards.



(a) $\lambda = 7.0$, amplitude = 30 mm (b) $\lambda = 0.5$, amplitude = 30 mm

Fig. 5. Forces estimated by the LNN+FNN combination and measured forces for two of the test trajectories [5].

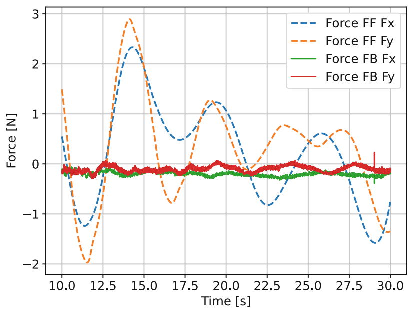
Fig. 5 illustrates the performance of the LNN+FNN combination. The force estimates for the low-frequent reference ($\lambda = 7.0$) show a clear offset. Mostly the forces are estimated much better like e.g. the example with the high-frequent reference ($\lambda = 0.5$). The offset may be caused by small differences in the initial configuration of the manipulator which can arise from the cables as mentioned before. To evaluate the force estimates, the MAE is computed after accounting for these offset. Then the average MAE for all 10 test datasets is 0.0505 N for the estimate from the neural networks and 0.0794 N for the white-box estimate.

3.6 Online Feedforward Control

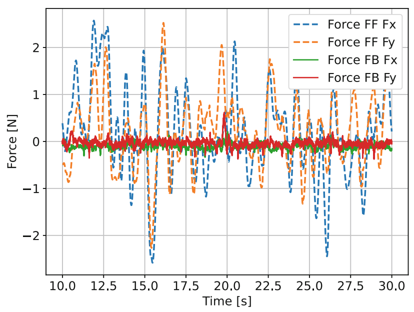
The ultimate goal is to improve the tracking accuracy of the manipulator by implementing the estimated forces as feedforward control inputs, i.e. "online". This performance is evaluated from a real-time experiment in which the estimated forces are added as feedforward control inputs (FF) to a standard closed-loop PD feedback controller (FB). In case of a perfect feedforward control, the estimated forces (FF) would result in perfect tracking of the reference path and the output of the feedback controller (FB) is zero due the absence of a tracking error.

Figure 6 presents the feedforward signals (FF) and the feedback signals (FB) recorded simultaneously during two of the experiments in which the feedforward control is estimated with the trained LNN+FNN combination. It can be seen that the contribution to the actuator force from the feedback controller (FB) is not zero, but it is significantly smaller than the estimated feedforward signal (FF) which apparently accounts for by far the largest part of the total forces required for the specified motion. This proves the effectiveness of the feedforward control estimated with the combined neural networks.

In this way the tracking accuracy is also improved considerably compared to using only feedback as shown in Fig. 7. In this figure also the results are included with the feedforward control from the white-box model. At first sight the latter approach tracks the reference even better, but a large part of the tracking error is the offset that has been discussed before in Sect. 3.5. Such offset can easily be



(a) $\lambda = 4.0$, amplitude = 30 mm.



(b) $\lambda = 0.5$, amplitude = 30 mm.

Fig. 6. Feedback (FB) and feedforward (FF from LNN+FNN) control inputs in closed-loop experiment with two different reference trajectories.

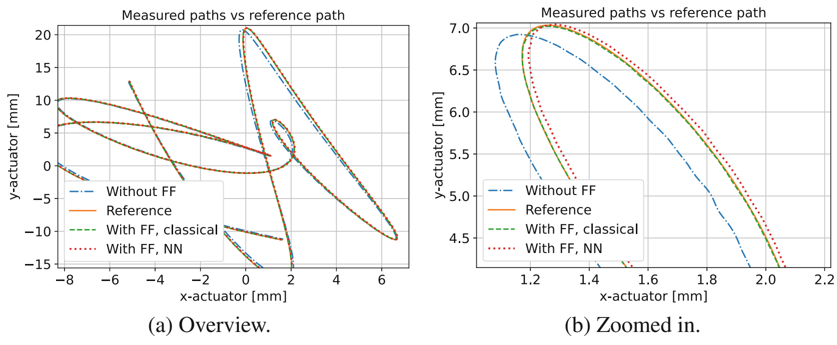


Fig. 7. Actuator motion during closed-loop experiment where the reference is generated with $\lambda = 4.0$, amplitude = 30 mm: Without FF as well as with FF from white-box model or LNN+FNN.

eliminated by using PID feedback control instead of PD control. From a more quantitative analysis it appeared that both feedforward control implementations reduce the tracking error with more than 90% compared to using only feedback control [5].

4 Conclusions

In this paper we demonstrate that the feedforward control generated by the successively trained LNN and FNN can be applied successfully for the considered 2-DOF manipulator with flexure joints. The LNN is trained first to ensure it accounts for the larger contribution of the conservative forces to the estimated actuator forces. In this way the advantages of constraining the network to a physical feasible solution are exploited. Only a rather small number of time samples is needed to train the LNN. The trained network can also handle new test trajectories which illustrates the robustness of the feedforward control. This robustness and a more general applicability of this approach will be investigated in the future.

References

1. Abdul-hadi, O.: Machine learning applications to robot control. PhD thesis, University of California, Berkeley (2018)
2. Agrawal, A., et al.: TensorFlow eager: a multi-stage, python-embedded DSL for machine learning. In: Talwalkar, A., Smith, V., Zaharia, M. (eds.) Proceedings of Machine Learning and Systems, vol. 1, 178–189 (2019)
3. Brouwer, D.M., Folkersma, K.G.P., Boer, S.E., Aarts, R.G.K.M.: Exact constraint design of a two-degree of freedom flexure-based mechanism. J. Mech. Rob. **5**(4), 041011 (2013)
4. Filip, S., Javeed, A., Trefethen, L.N.: Smooth random functions, random ODEs, and Gaussian processes. SIAM Rev. **61**(1), 185–205 (2019)

5. Heerze, E.: Data-driven feedforward control of a multi degree of freedom manipulator with flexure joints using machine learning. MSc thesis, University of Twente, Enschede (2021)
6. Khalil, W., Dombre, E.: Modeling. Identification and Control of Robots. Kogan Page Science, London (2002)
7. Liu, Z., Wang, B., Meng, Q., Chen, W., Tegmark, M., Liu, T.-Y.: Machine-learning nonconservative dynamics for new-physics detection. Phys. Rev. E **104**, 055302 (2021)
8. Lutter, M., Ritter, C., Peters, J.: Deep Lagrangian Networks: using physics as model prior for deep learning. In: 7th International Conference on Learning Representations (ICLR) (2019)
9. Lutter, M., Peters, J.: Combining physics and deep learning to learn continuous-time dynamics models. arXiv:2110.01894 (2023)
10. Wu, J., Wang, J., You, Z.: An overview of dynamic parameter identification of robots. Rob. Comput.-Integr. Manuf. **26**(5), 414–419 (2010)

On the Numerical Stability of Discretised Optimal Control Problems

Ashutosh Bijalwan[1] and José J. Muñoz[2](✉)

[1] Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE), Universitat Politècnica de Catalunya (UPC), Jordi Girona 31, 08034 Barcelona, Spain
abijalwan@cimne.upc.edu

[2] Laboratori de Càlcul Numèric (LaCàN), Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE), Institut de Matemàtiques de la UPC (IMTech), Universitat Politècnica de Catalunya (UPC), Jordi Girona 31, 08034 Barcelona, Spain
j.munoz@upc.edu

Abstract. Optimal Control Problems (OCP) consist in optimising an objective functional subjected to a set of Ordinary Differential Equations. In this work, we consider the effects on the stability of the numerical solution when this optimisation is discretised in time. In particular, we analyse a OCP with a quadratic functional and linear ODE, discretised with *Mid-point* and *implicit Euler*. We show that the numerical stability and the presence of numerical oscillations depends not only on the time-step size, but also on the parameters of the objective functional, which measures the amount of control input. Finally, we also show with an illustrative example that these results also carry over non-linear optimal control problems.

1 Introduction

Optimal control is a class of mathematical optimisation problems where the desired system state is determined by minimising/maximising a cost functional subjected to path constraints, written as ordinary differential equations (ODEs) and initial conditions. In real-world applications, close-form solutions of these problems are difficult to obtain and they are often solved numerically with non-linear programming techniques [4,5]. Based on the sequence of optimisation and time-discretisation, solution procedures for Optimal Control Problems (OCPs) can be classified as indirect and direct approaches. The indirect method first derives the necessary optimality conditions and forms the Differential-Algebraic Equations with the two-point boundary conditions, also known as the Two-Point Boundary Value Problem or Hamiltonian Boundary-Value Problem (HBVP), which are then discretised in time [11,12]. Conversely, the direct method first introduces the time-discretisation of the continuous system and then the necessary optimality conditions, derived from the resulting discrete system [2]. Both approaches have their unique properties and limitations, which have been described elsewhere, e.g. [5].

The stability of numerical integrators for Initial Value Problem (IVP) has been extensively studied. For instance, *explicit Euler (eE)* integrator is only conditionally stable whereas symplectic integrator viz. *implicit Euler (iE)* or *Mid-Point (MP)* are unconditionally stable schemes, which preserve first-integrals of motion [7,8]. Due to these promising properties of symplectic integrators, HBVP are frequently discretised

K. Nachbagauer and A. Held (Eds.): IUTAM 2022, IUTAM Bookseries 42, pp. 142–152, 2024.
https://doi.org/10.1007/978-3-031-50000-8_13

with MP or iE schemes [3,6,10]. However, due to the presence of two-point boundary conditions, implicit (symplectic) integration, which is unconditionally stable along the time-marching direction (forward direction), becomes explicit (conditionally stable) in the reverse direction, which the natural direction of the adjoint ODEs in OCP. Indeed symplectic partitioned Runge-Kutta methods (e.g. symplectic Euler (sE)) exploit this fact and discretise the state-adjoint system with implicit-explicit schemes, which leads to the cancellation of numerical error. This fact motivates the study in this work, where we show that originally stable schemes for ODEs, may become unstable in discretised OCP. Furthermore, we show that the stability and the presence of oscillations in the numerical solution depends on the magnitude of the control input. For simplicity and clarity on the exposition of our ideas, we have chosen to restrict our discussion to the numerical stability of MP, iE, and sE schemes applied to OCPs.

The chapter is structured as follows. In Sect. 1, we begin with a brief review of the available numerical methods for the OCP. In Sect. 2, we formally introduce the general framework of continuous OCP and describe the associated symplectic structure. Additionally, we present the time-discrete form of the HBVP. In Sect. 3, we employ the common MP, iE and sE schemes in an illustrative linear OCP, and provide a detailed analysis of the source of numerical oscillations and associated stability criteria. Finally, in Sect. 4, we generalise these ideas to prevent numerical oscillations in the non-linear OCPs. Conclusions are drawn in Sect. 5.

2 Optimal Control Problem: Indirect Method

Let us consider a continuous optimal control problem which seeks control $\boldsymbol{u}(t)$ and state $\boldsymbol{x}(t)$ trajectories that minimise an objective functional $\mathscr{J}(\boldsymbol{x},\boldsymbol{u})$ subjected to first-order IVP and equality constraints [2], i.e. are solution of the following optimisation problem

$$\begin{aligned} &\min_{\boldsymbol{x}(t),\,\boldsymbol{u}(t)} \mathscr{J}(\boldsymbol{x},\boldsymbol{u}) && (1)\\ &\text{s.t., } \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) - \dot{\boldsymbol{x}} = \boldsymbol{0} && \text{(State ODE)}\\ &\quad\;\; \boldsymbol{x}(0) - \boldsymbol{x}_0 = \boldsymbol{0} && \text{(Initial conditions)} \end{aligned}$$

where for clarity, we suppress time-dependence. A common choice for the functional $\mathscr{J}(\boldsymbol{x},\boldsymbol{u})$ in trajectory optimisation problems is to provide a measure of deviation of the state variable $\boldsymbol{x}(t)$ from a target state $\boldsymbol{x}_t$, and also add an associated input/control cost to achieve the desired state. Mathematically, the following form is usually employed,

$$\mathscr{J}(\boldsymbol{x},\boldsymbol{u}) := \int_0^T \left(\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_t)^{\mathsf{T}}\mathbf{R}(\boldsymbol{x}-\boldsymbol{x}_t) + \frac{\alpha}{2}\boldsymbol{u}^{\mathsf{T}}\mathbf{Q}\boldsymbol{u} \right) dt, \tag{2}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are the input and output matrices. Parameter $\alpha > 0$ regulates the amount of input control $\boldsymbol{u}$, and T is a fixed final time.

In order to deduce the optimality conditions of the constrained optimisation problem in Eq. (1), we introduce the Lagrange multipliers $\boldsymbol{\lambda}(t)$ and $\boldsymbol{\xi}$ and define the Lagrangian functional associated with problem (1) as

$$\mathscr{L}(\boldsymbol{x},\boldsymbol{u};\boldsymbol{\lambda},\boldsymbol{\xi}) = \mathscr{J}(\boldsymbol{x},\boldsymbol{u}) + \int_0^T \boldsymbol{\lambda}^\mathsf{T}(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) - \dot{\boldsymbol{x}})\,dt + \boldsymbol{\xi}^\mathsf{T}(\boldsymbol{x}(0) - \boldsymbol{x}_0). \tag{3}$$

After integrating by parts and rearranging the integrands, the Lagrangian becomes [4]

$$\mathscr{L}(\boldsymbol{x},\boldsymbol{u};\boldsymbol{\lambda},\boldsymbol{\xi}) = \int_0^T \left(\mathscr{H} + \dot{\boldsymbol{\lambda}}^\mathsf{T}\boldsymbol{x}\right) dt - \boldsymbol{\lambda}(T)^\mathsf{T}\boldsymbol{x}(T) + \boldsymbol{\lambda}(0)^\mathsf{T}\boldsymbol{x}(0) + \boldsymbol{\xi}^\mathsf{T}(\boldsymbol{x}(0) - \boldsymbol{x}_0)$$

where $\mathscr{H}(\boldsymbol{x},\boldsymbol{u};\boldsymbol{\lambda}) := \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_t)^\mathsf{T}\mathbf{R}(\boldsymbol{x}-\boldsymbol{x}_t) + \frac{\alpha}{2}\boldsymbol{u}^\mathsf{T}\mathbf{Q}\boldsymbol{u} + \boldsymbol{\lambda}^\mathsf{T}\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$ is the so-called Control Hamiltonian, which is conserved along optimal trajectories, provided $\frac{\partial\mathscr{H}}{\partial t} = 0$. First-order optimality conditions can be then derived from the derivatives of $\mathscr{L}$ with respect to $\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}$ and $\boldsymbol{\xi}$, which yields the following system of differential-algebraic equations with the two-point boundary conditions [3,6],

$$\dot{\boldsymbol{\lambda}} = -\nabla_{\boldsymbol{x}}\mathscr{H} \tag{4a}$$
$$\dot{\boldsymbol{x}} = \nabla_{\boldsymbol{\lambda}}\mathscr{H} \tag{4b}$$
$$\boldsymbol{0} = \nabla_{\boldsymbol{u}}\mathscr{H} \tag{4c}$$
$$\boldsymbol{x}(0) = \boldsymbol{x}_o,\ \boldsymbol{\lambda}(T) = \boldsymbol{0} \tag{4d}$$

The first three equations above are commonly named adjoint, state, and control-algebraic equations, respectively, and the system is known as the set of Euler-Lagrange equations (E-L), which together with the end conditions in (4d) constitutes the so-called Hamiltonian boundary-value problem (HBVP), with inherent symplectic structure [4].

We resort now to the numerical time integration of the HBVP by introducing a time-discretisation scheme of the continuous system in Eq. (4) with a uniform time-step size $\Delta t > 0$ ($n = 1, 2, \dots, N$), which we write in the following general form:

$$\frac{\boldsymbol{\lambda}_{n-1} - \boldsymbol{\lambda}_n}{\Delta t} = \mathbf{R}(\boldsymbol{x}_{n-\tau} - \boldsymbol{x}_t) + \nabla_{\boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}_{n-\tau}, \boldsymbol{u}_{n-\tau})^\mathsf{T}\boldsymbol{\lambda}_{n-\tau} \tag{5a}$$
$$\frac{\boldsymbol{x}_n - \boldsymbol{x}_{n-1}}{\Delta t} = \boldsymbol{f}(\boldsymbol{x}_{n-\tau}, \boldsymbol{u}_{n-\tau}) \tag{5b}$$
$$\boldsymbol{0} = \alpha\mathbf{Q}\boldsymbol{u}_{n-\tau} + \nabla_{\boldsymbol{u}}\boldsymbol{f}(\boldsymbol{x}_{n-\tau}, \boldsymbol{u}_{n-\tau})^\mathsf{T}\boldsymbol{\lambda}_{n-\tau} \tag{5c}$$
$$\boldsymbol{x}_0 = \boldsymbol{x}_o,\ \boldsymbol{\lambda}_N = \boldsymbol{0} \tag{5d}$$

with $\boldsymbol{x}_{n-\tau} := \tau\boldsymbol{x}_{n-1} + (1-\tau)\boldsymbol{x}_n$, and similarly for $\boldsymbol{u}_{n-\tau}$ and $\boldsymbol{\lambda}_{n-\tau}$. The values $\tau = 0$ and $\tau = \frac{1}{2}$ correspond to the well known implicit Euler (iE) and Mid-Point (MP) schemes, respectively.

3 Motivating Example: Linear OCP

Let us consider a propelled body (e.g. jellyfish) with mass m moving along $+y$ direction starting with an initial velocity v_o. There is a gravitational force with acceleration a acting along the $-y$ direction, in addition to a drag force with the form $f_d := bv$, with $b > 0$ a frictional coefficient. Furthermore, we assume that the jellyfish controls the propulsion velocity by regulating the fluid ejection along the $-y$ direction resulting in a thrust force u along $+y$ direction. Linear momentum balance along $+y$ direction results in the following state IVP,

$$\begin{aligned} f(v,u) - \dot{v} &= 0 \\ v(0) - v_o &= 0 \end{aligned} \tag{6}$$

where $f(v,u) := -\frac{b}{m}v + \frac{u}{m} - a$. We are interested in obtaining a control function $u(t)$, which drives the jellyfish from initial velocity v_o to a target velocity v_t in a given time T. Essentially, we want to minimise the following cost functional

$$\mathscr{J}(v,u) := \int_0^T \left(\frac{1}{2}(v - v_t)^2 + \frac{\alpha}{2}u^2 \right) dt \tag{7}$$

subject to the IVP in Eq. (6). A closed-form solution of the posed problem is given by

$$v(t) = C_1 \left(\frac{b}{m} - \gamma \right) e^{\gamma t} + C_2 \left(\frac{b}{m} + \gamma \right) e^{-\gamma t} + v_p \tag{8a}$$

$$\lambda(t) = C_1 e^{\gamma t} + C_2 e^{-\gamma t} + \lambda_p \tag{8b}$$

$$u(t) = \frac{\lambda(t)}{\alpha m} \tag{8c}$$

where

$$\gamma = \sqrt{\frac{b^2}{m^2} + \frac{1}{\alpha m^2}},\ v_p = \frac{v_t - \alpha bma}{\alpha m^2 \gamma^2},\ \lambda_p = -\frac{bv_t + ma}{m\gamma^2},$$
$$C_1 = \frac{m(v_o - v_p)e^{-\gamma T} + (b + m\gamma)\lambda_p}{(b - m\gamma)e^{-\gamma T} - (b + m\gamma)e^{\gamma T}},\ C_2 = \frac{m(v_o - v_p) - (b - m\gamma)C_1}{b + m\gamma}$$

Irrespective of the value of $\alpha > 0$, we have that $\gamma > 0$, and one would expect a non-oscillatory state, adjoint and control functions. We will show in the next sections that the numerical solution of the OCP with the MP and iE time-discretisation schemes is not necessarily non-oscillatory.

3.1 Mid-Point Scheme

By using the value $\tau = 1/2$ in Eq. (5), a mid-point (MP) time-discretisation of the ODE system is obtained,

$$\frac{\lambda_n - \lambda_{n-1}}{\Delta t} - \frac{b}{m}\lambda_{n-\frac{1}{2}} + v_{n-\frac{1}{2}} - v_t = 0, \tag{9a}$$

$$\frac{v_n - v_{n-1}}{\Delta t} + \frac{b}{m}v_{n-\frac{1}{2}} - \frac{1}{m}u_{n-\frac{1}{2}} + a = 0, \tag{9b}$$

$$u_{n-\frac{1}{2}} + \frac{1}{\alpha m}\lambda_{n-\frac{1}{2}} = 0. \tag{9c}$$

After substituting the boundary conditions $v_0 = v_o$ and $\lambda_N = 0$, the resulting linear system of equations can be solved with conventional linear solvers.

In order to study the stability of the MP scheme, it will be helpful to express the system with the independent variables (v and λ). Notice that the control equation in Eq. (9c) is linear, so that replacing Eq. (9c) into Eq. (9b), with the definitions

$$p := \frac{b}{m} + \frac{2}{\Delta t};\ q := \frac{b}{m} - \frac{2}{\Delta t};\ s := \frac{1}{\alpha m^2} \tag{10}$$

results in the following reduced discrete system

$$q\lambda_n + p\lambda_{n-1} - v_{n-1} - v_n + 2v_t = 0 \tag{11a}$$
$$pv_n + qv_{n-1} + s\lambda_{n-1} + s\lambda_n + 2a = 0 \tag{11b}$$

By defining the vector $\boldsymbol{z}_n := \{v_n, \lambda_n\}^T$, the MP scheme can be expressed as

$$\boldsymbol{z}_n = \mathbf{A}\boldsymbol{z}_{n-1} + \boldsymbol{a} \tag{12}$$

where $\mathbf{A}$ is the amplification matrix [1,9] and $\boldsymbol{a}$ a constant vector. In the present case they have the form

$$\mathbf{A} := \frac{-1}{s+pq}\begin{bmatrix} s+q^2 & -s(p-q) \\ -(p-q) & s+p^2 \end{bmatrix};\ \boldsymbol{a} := \frac{-2}{s+pq}\begin{Bmatrix} qa - sv_t \\ a + pv_t \end{Bmatrix}.$$

The eigenvalues e_1 and e_2 of matrix $\mathbf{A}$ are real and distinct, and are given by

$$e_{1,2} = \left\{\frac{2+\gamma\Delta t}{2-\gamma\Delta t},\ \frac{2-\gamma\Delta t}{2+\gamma\Delta t}\right\}. \tag{13}$$

and the spectral radius of $\mathbf{A}$ is

$$\rho(\mathbf{A}) = \frac{2+\gamma\Delta t}{|2-\gamma\Delta t|}. \tag{14}$$

Consequently, the time-discretisation scheme is stable if $\rho(\mathbf{A}) \leq 1$. It seems that there exist no admissible pair $(\gamma, \Delta t) \in \mathbb{R}^+$ for which $\rho(\mathbf{A}) \leq 1$. However, if we impose

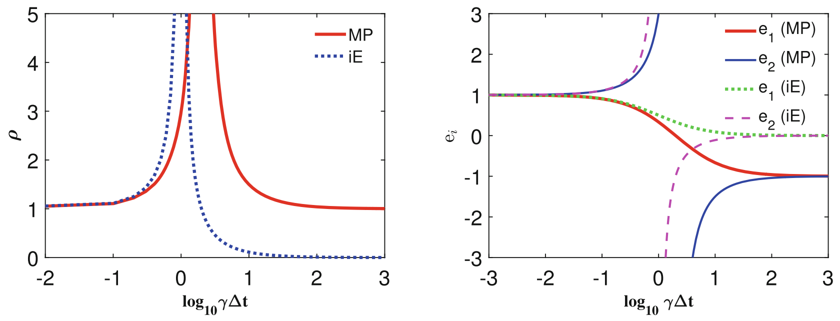


Fig. 1. Mid-point (MP) and implicit Euler (iE) scheme: (a) spectral radius, and (b) eigenvalues.

restrictions on $\gamma\Delta t$ such that $\log_{10}(\gamma\Delta t/2) \in (-\infty,-\varepsilon)\cup(\varepsilon,\infty)$ with $\varepsilon \approx 2$, numerical evidence shows that the solution remains bounded.

In order to analyse the presence of numerical oscillations, we examine the value of α for which the eigenvalues change their sign. The eigenvalues in Eq. (13) switch sign from negative to positive if $\gamma^2 - \frac{4}{\Delta t^2} < 0$ and for the threshold value $\gamma = \frac{2}{\Delta t}$, the system blows up. In summary,

$$e_i = \begin{cases} < 0, & \alpha < \alpha_{th,MP} \quad \text{(Oscillatory response)} \\ > 0, & \alpha > \alpha_{th,MP} \quad \text{(No oscillations)} \end{cases} \tag{15}$$

with

$$\alpha_{th,MP} := \frac{\Delta t^2}{4m^2 - b^2\Delta t^2}. \tag{16}$$

3.2 Implicit Euler Scheme

By using the value $\tau = 0$ in Eq. (5), Implicit Euler (iE) time-discretisation results in the following discrete system

$$\frac{\lambda_n - \lambda_{n-1}}{\Delta t} - \frac{b}{m}\lambda_n + v_n - v_t = 0, \tag{17a}$$

$$\frac{v_n - v_{n-1}}{\Delta t} + \frac{b}{m}v_n - \frac{1}{m}u_n + a = 0, \tag{17b}$$

$$u_n + \frac{1}{\alpha m}\lambda_n = 0. \tag{17c}$$

After substituting the boundary conditions $v_0 = v_o$ and $\lambda_N = 0$, the solution of the resulting linear system can be obtained.

In order to study the stability of the iE scheme, we express our system in terms of the independent variables v and λ. Substituting Eq. (17c) into Eq. (17b), and with the new definitions

$$p^* := \frac{b}{m} + \frac{1}{\Delta t};\ q^* := \frac{b}{m} - \frac{1}{\Delta t};\ r := \frac{1}{\Delta t};\ s := \frac{1}{\alpha m^2} \tag{18}$$

the system in (17) is equivalent to

$$q^*\lambda_n + r\lambda_{n-1} - v_n + v_t = 0 \tag{19}$$

$$p^*v_n - rv_{n-1} + s\lambda_n + a = 0. \tag{20}$$

or in terms of the vector $\boldsymbol{z}_n := \{v_n, \lambda_n\}^T$, it can be expressed as

$$\boldsymbol{z}_n = \mathbf{B}\boldsymbol{z}_{n-1} + \boldsymbol{b} \tag{21}$$

with

$$\mathbf{B} := \frac{-r}{s+p^*q^*}\begin{bmatrix} -q^* & -s \\ -1 & p^* \end{bmatrix},\ \boldsymbol{b} := \frac{-1}{s+p^*q^*}\begin{Bmatrix} q^*a - sv_t \\ a + p^*v_t \end{Bmatrix}.$$

The eigenvalues of matrix $\mathbf{B}$ and its spectral radius $\rho(\mathbf{B})$ are given by

$$e_{1,2} = \left\{ \frac{1}{1+\gamma\Delta t}, \frac{1}{1-\gamma\Delta t} \right\},\ \rho(\mathbf{B}) = \frac{1}{|1-\gamma\Delta t|} \tag{22}$$

The time-discretisation scheme will be thus stable if $|\gamma\Delta t - 1| \geq 1$. Since $\gamma\Delta t \in \mathbb{R}^+$, we are left with the restriction $\gamma \geq \frac{2}{\Delta t}$.

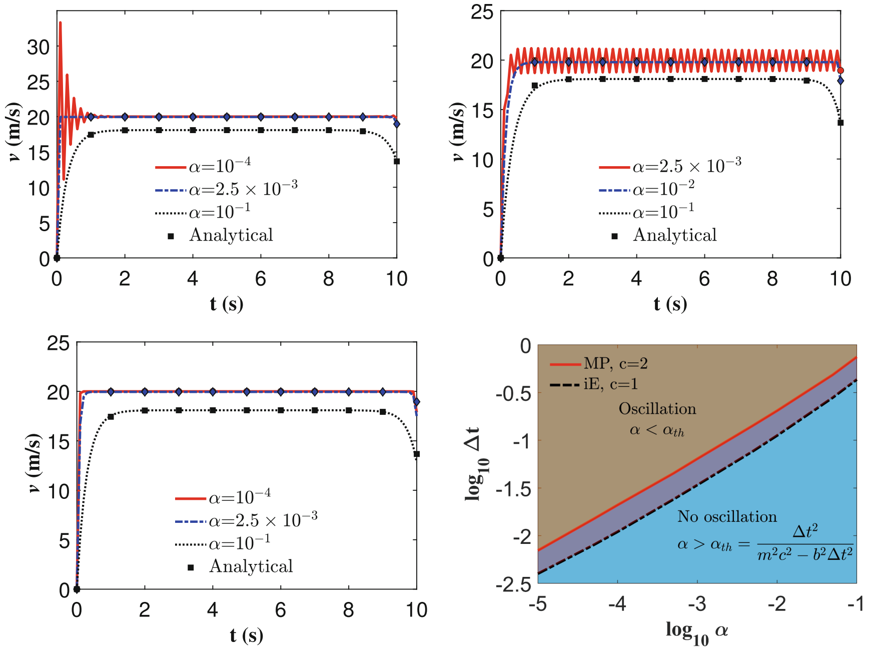


Fig. 2. State optimal velocity: (a) MP scheme and (b) iE scheme, (c) sE scheme (dots: Analytical solution), and (d) phase diagram $(\alpha, \Delta t)$ for MP and iE schemes.

A similar analysis to the MP scheme indicates that the change of sign in e_2 in the iE scheme occurs when $1-\gamma\Delta t$ changes sign, and for the threshold value of $\gamma = \frac{1}{\Delta t}$ the system blows out. Summarising,

$$\frac{1}{1-\gamma\Delta t} = \begin{cases} < 0, & \alpha < \alpha_{th,iE} \text{ (Oscillatory response)} \\ > 0, & \alpha > \alpha_{th,iE} \text{ (No oscillations)} \end{cases} \tag{23}$$

with

$$\alpha_{th,iE} := \frac{\Delta t^2}{m^2 - b^2\Delta t^2}. \tag{24}$$

We note that from the expressions in (16) and (24), it follows that as friction is reduced (i.e. b decreases), the oscillations may appear for small mass m or large time-step Δt. Instead, large friction values will prevent the presence of numerical oscillations.

3.3 Symplectic Euler Scheme

By using the value $\tau = 0$ for adjoint and $\tau = 1$ for state in Eq. (5), symplectic Euler (sE) time-discretisation results in the following discrete system

$$\frac{\lambda_n - \lambda_{n-1}}{\Delta t} - \frac{b}{m}\lambda_n + v_{n-1} - v_t = 0, \tag{25a}$$

$$\frac{v_n - v_{n-1}}{\Delta t} + \frac{b}{m}v_{n-1} - \frac{1}{m}u_n + a = 0, \tag{25b}$$

$$u_n + \frac{1}{\alpha m}\lambda_n = 0. \tag{25c}$$

After substituting the boundary conditions $v_0 = v_o$ and $\lambda_N = 0$, the solution of the resulting linear system can be obtained.

In order to study the stability of the sE scheme, we express our system in terms of the independent variables v and λ. Substituting Eq. (25c) into Eq. (25b), and with previous definitions the system in (25) is equivalent to

$$q^*\lambda_n + r\lambda_{n-1} - v_{n-1} + v_t = 0 \tag{26}$$

$$rv_n + q^*v_{n-1} + s\lambda_n + a = 0. \tag{27}$$

or in terms of the vector $\boldsymbol{z}_n := \{v_n, \lambda_n\}^T$, it can be expressed as

$$\boldsymbol{z}_n = \mathbf{C}\boldsymbol{z}_{n-1} + \boldsymbol{c} \tag{28}$$

with

$$\mathbf{C} := \frac{-1}{rq^*}\begin{bmatrix} s + q^{*2} & -rs \\ -r & r^2 \end{bmatrix}, \ \boldsymbol{c} := \frac{-1}{rq^*}\begin{Bmatrix} q^*a - sv_t \\ rv_t \end{Bmatrix}.$$

The eigenvalues of matrix $\mathbf{C}$ are given by

$$e_{1,2} = \frac{1}{\varphi}\left\{\gamma^2 + \varphi \pm \gamma\sqrt{\gamma^2 + 2\varphi}\right\} \tag{29}$$

with $\varphi := \frac{2}{\Delta t}(\frac{1}{\Delta t} - \frac{b}{m})$.

If $\Delta t < 1$, and $\frac{b}{m} \leq 1$, which is generally true, eigenvalues are always positive and the sE scheme result in non-oscillatory optimal trajectories.

We have numerically verified the theoretical results and thresholds in (16) and (24) by using the values $(m, b, a, v_o, v_t, T, \Delta t) = (1, 1, 1, 0, 20, 10, 0.1)$, which imply the values $\alpha_{th,MP} = 2.5 \times 10^{-3}$ and $\alpha_{th,iE} = 1.01 \times 10^{-2}$. Reducing α below these values, numerical oscillations are obtained, in agreement with conditions in (15) and (23), as shown in Fig. 2. Theoretical results for the spectral radius and eigenvalues are shown in Fig. 1. We point out that there is a region where both schemes become unstable and that stable results are obtained if we are sufficiently far from this region. As expected, for the sE scheme, eigenvalues remain strictly positive and result in non-oscillatory trajectories irrespective of α.

For verifying the thresholds of α_{th}, we generate the stability envelope from numerical experiments in Fig. 2. We found close matches with Eq. (16) and Eq. (24). We remark that the phase boundary of the MP scheme lies above the iE scheme and shows numerical oscillations much later than the iE scheme.

4 Nonlinear OCP: Inverted Elastic Pendulum

In the second application, we consider a planar elastic inverted pendulum consisting of two point masses m_1 and m_2 and linked by an elastic spring. The system is subjected to a gravitational force field along $-y$ direction with intensity a. Position $\boldsymbol{x}_1$ of m_1 is constrained to move along the x-axis, while m_2 is free to move in x-y plane. The initial positions of m_1 and m_2 are respectively $(0,0)$ and $(0.3,1)$ (see Fig. 3). We assume that the spring potential energy varies quadratically with the length increment from a rest-length l_o, i.e. $U(\boldsymbol{x}) := \frac{k}{2}(l(\boldsymbol{x}) - l_o)^2$. Linear momentum balance can be expressed as

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) - \dot{\boldsymbol{v}} &= \boldsymbol{0} \\ \boldsymbol{v} - \dot{\boldsymbol{x}} &= \boldsymbol{0} \\ \boldsymbol{x}(0) - \boldsymbol{x}_o &= \boldsymbol{0} \\ \boldsymbol{v}(0) - \boldsymbol{v}_o &= \boldsymbol{0} \end{aligned} \tag{30}$$

where $l = \|\boldsymbol{x}_2 - \boldsymbol{x}_1\|_2$, $\boldsymbol{x} = \{\boldsymbol{x}_1,\, \boldsymbol{x}_2\}^\mathsf{T}$, $\boldsymbol{v} = \{\boldsymbol{v}_1, \boldsymbol{v}_2\}^\mathsf{T}$, $\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) := -\mathbf{M}^{-1}\left(\nabla_{\boldsymbol{x}} U + \boldsymbol{a}\right)$, $\mathbf{M} := \mathrm{diag}(m_1, m_1, m_2, m_2)$, and $\boldsymbol{a} := \{0, a, 0, a\}^\mathsf{T}$. We are interested in finding the control velocity $u = v_1$ which stabilises the system in upright configuration. Equivalently, we aim at minimising the cost functional

$$\mathscr{J}(\boldsymbol{x},\boldsymbol{u}) := \int_0^T \left(\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_t)^\mathsf{T}\mathbf{R}(\boldsymbol{x}-\boldsymbol{x}_t) + \frac{\alpha}{2}\boldsymbol{u}^\mathsf{T}\mathbf{Q}\boldsymbol{u}\right) dt \tag{31}$$

subjected to Eq. (30), where $\boldsymbol{u} = \boldsymbol{v}$, $\mathbf{Q} = \mathrm{diag}(1,0,0,0)$, $\mathbf{R} = \mathrm{diag}(0,0,0,1)$, and $\boldsymbol{x}_t = \{0,0,0,x_t\}^\mathsf{T}$.

For the numerical test, we assign system parameters $(m_1, m_2, k, a, x_t, T, \Delta t) = (1, 1, 10, 0.1, 2, 4, 0.2)$, and we use $\alpha \in (10^{-4}, 10^{-2})$. Next, we discretise our system with MP scheme, i.e. Eq. (5) with $\tau = \frac{1}{2}$). The resulting system of non-linear equations is numerically solved. Based on the linear analysis, we observed that for the MP scheme with small Δt, we have that $\alpha_{th} \sim \mathscr{O}(\Delta t^2)$, hereby coined as a *conservative stability criteria*, and reducing α below this value should trigger numerical oscillations. Figure 3 shows the horizontal position of mass 1 with three α values. It can be seen that for $\alpha = 10^{-2}$ we have a smooth trajectory, with $\alpha = 10^{-3}$ we incubate a small kink, and $\alpha = 10^{-4}$ results in an oscillating optimal trajectory. For comparison, we have solved the above system with the sE scheme and numerical results shows the stable state trajectory (see Fig. 3) and is consistent with the theoretical findings of the linear OCP.

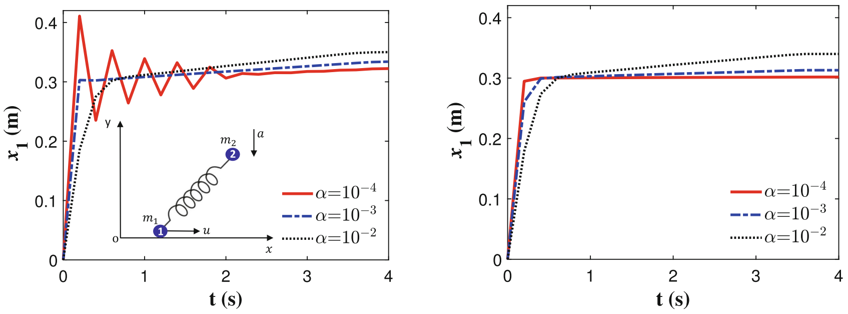


Fig. 3. Inverted pendulum: mass m_1 horizontal displacement with α (a) MP scheme, and (b) sE scheme.

5 Summary

We have shown that the stability criteria in OCP depends not only on the parameters of the ODEs and time-step of the discretisation, but also on the parameters of the objective function. Furthermore, small values of the parameters in the control cost function may also induce numerical oscillations. We have demonstrated the origin of these instabilities and oscillations for a linear problem, and also illustrated numerically how these ideas also carry over problems with non-linear ODEs. Our numerical experiments suggest that as α diminishes, we must reduce the time-step size to circumvent numerical oscillations in optimal trajectories, i.e., $\alpha \sim \mathcal{O}(\Delta t^2)$.

Acknowledgements. This work is financially supported by the Spanish Ministry of Science and Innovation, under Severo Ochoa program CEX2018-000797-S, and the research project DynAd2, with reference PID2020-116141GB-I00.

References

1. Asher, U.M., Petzold, L.R.: Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations. Society for Industrial and Applied Mathematics (SIAM), (1998)
2. Betts, J.T.: Practical Methods for Optimal Control and Estimation Using Nonlinear Programming. Society for Industrial and Applied Mathematics (SIAM) (2010)
3. Betsch, P., Becker, C.: Conservation of generalized momentum maps in mechanical optimal control problems with symmetry. Int. J. Numer. Meth. Eng. **111**(2), 144–175 (2017). https://doi.org/10.1002/nme.54594
4. Bryson, A.E.; Ho, Y.C.: Applied Optimal Control. Optimization, Estimation and Control. Taylor & Francis (1975)
5. Conway, B.A.: A survey of methods available for the numerical optimization of continuous dynamic systems. J. Optim. Theory Appl. **152**(2), 271–306 (2012). https://doi.org/10.1007/s10957-011-9918-z
6. Flaßkamp, K., Murphey, T.D.: Structure-preserving local optimal control of mechanical systems. Optimal Contr. Appl. Methods **40**(2), 310–329 (2019). https://doi.org/10.1002/oca.2479

7. Gonzalez, O.: Mechanical systems subject to holonomic constraints: differential algebraic formulations and conservative integration. Physica D **132**(1–2), 165–174 (1999). https://doi.org/10.1016/S0167-2789(99)00054-8
8. Hairer, E., Wanner, G., Lubich, C.: Symplectic integration of hamiltonian systems. In Geometric Numerical Integration. Springer, Berlin, Heidelberg (2006)
9. Hilber, H.M., Hughes, T.J.R.: Collocation, dissipation and [overshoot] for time integration schemes in structural dynamics. Int. J. Num. Meth. Engin. **6**(1), 99–117 (1978)
10. Koch, M.W., Leyendecker, S.: Energy momentum consistent force formulation for the optimal control of multibody systems. Multibody Sys.Dyn. **29**(4), 381–401 (2013). https://doi.org/10.1007/s11044-012-9332-9
11. Miller, M.I., Trouvé, A., Younes, L.: Hamiltonian systems and optimal control in computational anatomy: 100 years since d'arcy thompson. Annu. Rev. Biomed. Eng. **17**, 447–509 (2015). https://doi.org/10.1146/annurev-bioeng-071114-040601
12. Sharp, J.A., Burrage, K., Simpson, M.J.: Implementation and acceleration of optimal control for systems biology. J. R. Soc. Interface **18**(181), 20210241 (2021). https://doi.org/10.1098/rsif.2021.0241

On the Usage of Analytically Computed Adjoint Gradients in a Direct Optimization for Time-Optimal Control Problems

Daniel Lichtenecker[1](✉), Philipp Eichmeir[2], and Karin Nachbagauer[2,3]

[1] Technical University of Munich, Germany; TUM School of Engineering and Design, Department of Mechanical Engineering, Chair of Applied Mechanics, Munich Institute of Robotics and Machine Intelligence (MIRMI), Boltzmannstraße 15, 85748 Garching, Germany

daniel.lichtenecker@tum.de

[2] University of Applied Sciences Upper Austria, Campus Wels, Stelzhamerstraße 23, 4600 Wels, Austria

{philipp.eichmeir,karin.nachbagauer}@fh-wels.at

[3] Institute for Advanced Study, Technical University of Munich, Lichtenbergstraße 2a, 85748 Garching, Germany

Abstract. This paper discusses time-optimal control problems and describes a workflow for the use of analytically computed adjoint gradients considering a discrete control parameterization. The adjoint gradients are used here to support a direct optimization method, such as Sequential Quadratic Programming (SQP), by providing analytically computed gradients and avoiding the elaborate numerical differentiation. In addition, the adjoint variables can be used to evaluate the necessary first-order optimality conditions regarding the Hamiltonian function and gives an opportunity to discuss the sensitivity of a solution with respect to the refinement of the discretization of the control. To further emphasize the advantages of adjoint gradients, there is also a discussion of the structure of analytical gradients computed by a direct differentiation method, and the difference in the dimensions compared to the adjoint approach is addressed. An example of trajectory planning for a robot shows application scenarios for the adjoint variables in a cubic spline parameterized control.

1 Introduction

Optimal control theory is based on the calculus of variations and deals with finding optimal trajectories for nonlinear dynamical systems, e.g. spacecrafts or multibody systems like robots. The works by Kelley [4] and Bryson and Ho [1] have to be mentioned as groundbreaking in the field of optimal control theory and serve as basis for extensive subsequent research, also in the field of time-optimal control.

As a special class of time-optimal control problems considering final constraints, one can cite the control of a robot arm designed in such a way that the operation time for a rest-to-rest maneuver becomes minimal. Following an indirect optimization approach, such problems can be transformed into a two-point boundary value problem, which can usually be solved by shooting or full collocation methods. Alternatively, a direct optimization approach can be pursued, in which the boundary value problem is posed as a nonlinear programming problem method, see e.g. [12] for the time-optimal trajectory planning considering the continuity required to respect technological limits of real robots.

An alternative to the mentioned methods is offered by indirect gradient methods, which are considered to be particularly robust with respect to initial controls. The work by Bryson and Ho [1] shows how the gradient can be computed straightforward using adjoint variables. With this gradient information optimal control problems can be solved iteratively by the use of nonlinear optimization routines, as described in the sense of optimal control or parameter identification in multibody systems e.g. in [8]. The work by Eichmeir et al. [2] extends the theory for time-optimal control problems to dynamic systems under final constraints. Such problems arise e.g. in space vehicle dynamics during minimum time moon ascending/descending maneuvers or in robotics in the case that the time for a rest-to-rest maneuver should become a minimum. Such problems can be considered as two-point boundary value problems, with the major drawback of requiring an initial guess close to the optimal solution. Otherwise, the optimal control problem could be solved via the adjoint method which is an efficient way to compute the direction of the steepest descent of a cost functional. However, when using such indirect methods to solve optimal control problems, a major drawback appears in the computation of the Hamiltonian and the required derivatives: they may be complex and furthermore need to be recomputed often during the simulation. Moreover, depending on the variables or parameters to be identified in the optimal control strategy, it is difficult to assign a physical meaning to the adjoint variables.

This paper focuses on solving time-optimal control problems with a classical direct optimization method and then evaluating the respective optimality conditions based on an indirect optimization approach. The adjoint variables can be investigated to efficiently compute the gradients of the cost functional and the constraints. Moreover, the adjoint variables can be investigated to exploit the optimality conditions regarding the Hamiltonian function. To demonstrate the use of analytically computed adjoint gradients, the time-optimal trajectory planning of a Selective Compliance Assembly Robot Arm (SCARA) is solved by an SQP method and the optimality conditions regarding the Hamiltonian function are evaluated by the adjoints. The application shows the easy access to the adjoint gradients and discusses the latter mentioned role of the adjoint variables in the optimality conditions.

2 Use of Adjoint Variables in Direct Optimization Approaches

The aim of this paper is to determine a control $\mathbf{u}(t) = \mathbf{u}^*$ and a final time $t_f = t_f^*$ of a dynamical system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \quad \text{with} \quad \mathbf{x}(0) = \mathbf{x}_0, \tag{1}$$

such that the scalar performance measure

$$J(\mathbf{x}(t), \mathbf{u}(t), t_f) = \int_{t_0}^{t_f} \Big[1 + P(\mathbf{x}(t), \mathbf{u}(t))\Big]\, \mathrm{d}t \tag{2}$$

becomes a minimum with respect to a final constraint

$$\boldsymbol{\phi}(\mathbf{x}(t_f), t_f) = \mathbf{0} \in \mathbb{R}^q. \tag{3}$$

Inequality constraints on the state $\mathbf{x} \in \mathbb{R}^n$ and the control $\mathbf{u} \in \mathbb{R}^m$ are considered by the scalar penalty function P. To be specific, violations of inequality constraints within the time interval $t \in [t_0, t_f]$ are taken into account as an additional term in the cost functional in Eq. (2). The above optimal control problem can generally be solved by a direct or indirect optimization approach. In this paper, the original infinite dimensional optimization problem is transformed into a finite dimensional one by parameterizing the control with a finite set of optimization variables $\mathbf{z} \in \mathbb{R}^z$ including the final time and the control parameterization. Thus, the resulting nonlinear programming (NLP) problem can be solved with classical direct optimization approaches such as the SQP method [9].

How to Interpret the Results from a Direct Optimization Algorithm An optimal point $\mathbf{z} = \mathbf{z}^*$ fulfills the well-known Karush-Kuhn-Tucker (KKT) conditions [3,5], but these conditions do not provide any information about the quality of the control parameterization with respect to the infinite dimensional optimization problem. The basic idea to interpret an optimal point $\mathbf{z}^*$ is to relate the direct optimization approach to Pontryagin's minimum principle [11]. The optimality conditions based on an indirect optimization approach [2] can be used for this idea. Figure 1 illustrates a rough flowchart for the interpretation of results obtained by a direct optimization approach. This approach requires the Hamiltonian of the system to evaluate Pontryagin's minimum principle. The Hamiltonian for time-optimal control problems related to the cost functional in Eq. (2) can be formulated as

$$\mathcal{H}(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\lambda}(t)) := 1 + P(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}(t)^\mathsf{T} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{4}$$

in which the multiplier $\boldsymbol{\lambda}(t) = \mathbf{p}(t) + \mathbf{R}(t)\boldsymbol{\nu}$ is computed by a linear combination of the adjoint variables $\mathbf{p} \in \mathbb{R}^n$ and $\mathbf{R} \in \mathbb{R}^{n\times q}$. The vector $\boldsymbol{\nu} \in \mathbb{R}^q$ is a multiplier to combine both adjoint variables. A deep insight into the combination of both adjoint variables is presented in [2]. The Hamiltonian in Eq. (4) is used in Sect. 4 to interpret the results of a time-optimal control problem obtained by a direct optimization approach as depicted in Fig. 1.

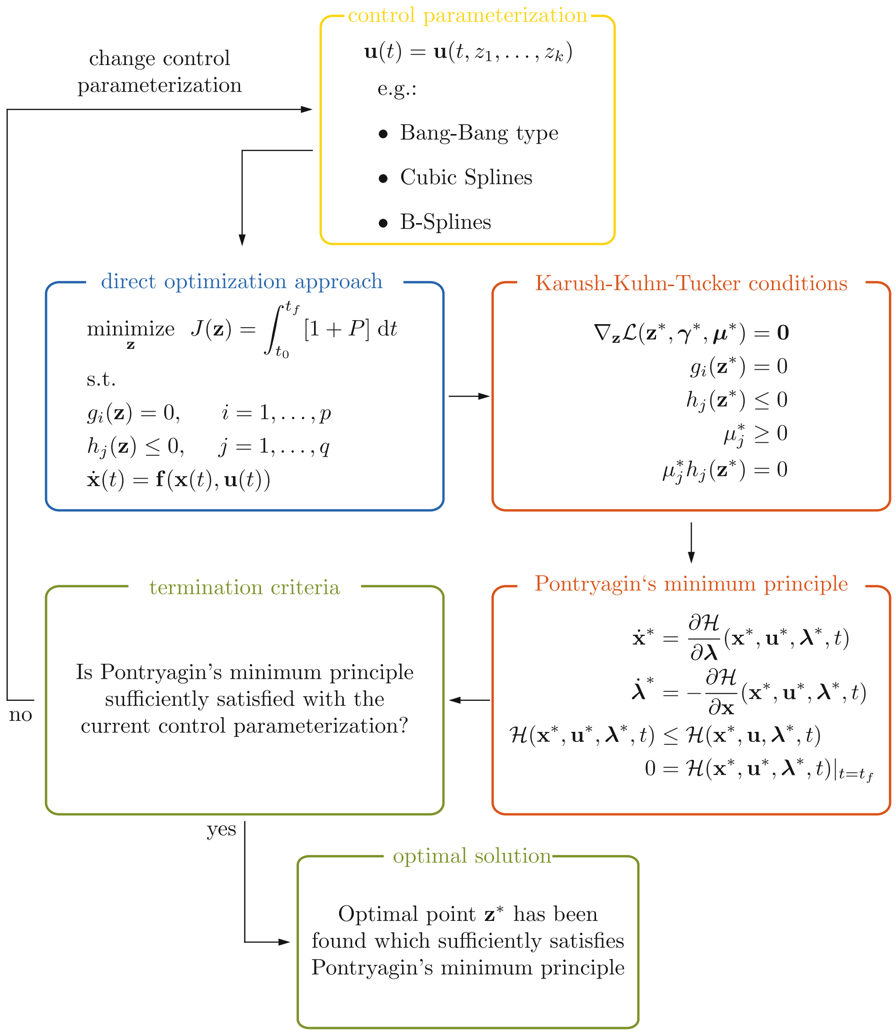

control parameterization
$\mathbf{u}(t) = \mathbf{u}(t, z_1, \ldots, z_k)$
e.g.:
• Bang-Bang type
• Cubic Splines
• B-Splines
change control parameterization
direct optimization approach
$\underset{\mathbf{z}}{\text{minimize}} \quad J(\mathbf{z}) = \int_{t_0}^{t_f} [1 + P] \, \mathrm{d}t$
s.t.
$g_i(\mathbf{z}) = 0, \quad i = 1, \ldots, p$
$h_j(\mathbf{z}) \leq 0, \quad j = 1, \ldots, q$
$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$
Karush-Kuhn-Tucker conditions
$\nabla_{\mathbf{z}} \mathcal{L}(\mathbf{z}^*, \boldsymbol{\gamma}^*, \boldsymbol{\mu}^*) = \mathbf{0}$
$g_i(\mathbf{z}^*) = 0$
$h_j(\mathbf{z}^*) \leq 0$
$\mu_j^* \geq 0$
$\mu_j^* h_j(\mathbf{z}^*) = 0$
termination criteria
Is Pontryagin's minimum principle sufficiently satisfied with the current control parameterization?
no
Pontryagin's minimum principle
$\dot{\mathbf{x}}^* = \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, t)$
$\dot{\boldsymbol{\lambda}}^* = -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, t)$
$\mathcal{H}(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, t) \leq \mathcal{H}(\mathbf{x}^*, \mathbf{u}, \boldsymbol{\lambda}^*, t)$
$0 = \mathcal{H}(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, t)|_{t=t_f}$
yes
optimal solution
Optimal point $\mathbf{z}^*$ has been found which sufficiently satisfies Pontryagin's minimum principle

Fig. 1. Flowchart to interpret the results from a direct optimization algorithm with Pontryagin's minimum principle

3 Computation of First-Order Derivatives

Classical gradient-based optimization algorithms rely on the derivatives of the cost functional and the constraints with respect to the optimization variables $\mathbf{z}$. The computation of these gradients takes a key role in such optimization algorithms and the convergence of the optimization depends on the accuracy of the gradients. In addition to accuracy, efficient computation of gradients is especially important for large numbers of optimization variables. Thus, the computational effort to solve the optimization problem depends significantly on the efficient computation of gradients. Figure 2 summarizes the most common approaches for the computation of first-order gradients. The *finite differences* method is the easiest approach to code, but suffers in terms of computational effort especially for a large number of optimization variables. In case of using (forward or backward) finite differences, the state equations have to be solved $(1+z)$ times in order to evaluate the numerical gradients with respect to z optimization variables. Thus, the number of forward simulations grows linearly with the number of optimization variables. In contrast to this numerical approach, the *direct differentiation* and the *adjoint method* are referred as analytical approaches to compute gradients. Both approaches lead to exact gradient information and using them in an optimization scheme leads to an increase in efficiency. The characteristics of the analytical approaches are discussed in the following sections.

3.1 Direct Differentiation Approach for Discrete Control Parameterization

The direct differentiation approach is based on the sensitivity of the state equations and is briefly discussed in this section. In this paper, the control is described by $\mathbf{u}(t) = \mathbf{C}\,\bar{\mathbf{u}}$, in which the vector $\bar{\mathbf{u}}^\mathsf{T} = \left(\hat{\mathbf{u}}_1^\mathsf{T}, \ldots, \hat{\mathbf{u}}_m^\mathsf{T}\right) \in \mathbb{R}^{m\cdot k}$ collects k grid nodes of the m equidistant time-discretized controls and the matrix $\mathbf{C}(t) \in \mathbb{R}^{m\times m\cdot k}$ maps the grid nodes to a time dependent function. The interpolation matrix $\mathbf{C}$ has to be determined once *a priori* and depends on the chosen interpolation order [6].

By using this control parameterization, the gradient of the cost functional is directly obtained by differentiating it with respect to the grid nodes as

$$\nabla_{\bar{\mathbf{u}}} J^\mathsf{T} = \int_{t_0}^{t_f} \left[\frac{\partial P}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial \bar{\mathbf{u}}} + \frac{\partial P}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \bar{\mathbf{u}}}\right] \mathrm{d}t \tag{5}$$

$$= \int_{t_0}^{t_f} \left[P_\mathbf{x}\mathbf{x}_{\bar{\mathbf{u}}} + P_\mathbf{u}\mathbf{C}\right] \mathrm{d}t, \tag{6}$$

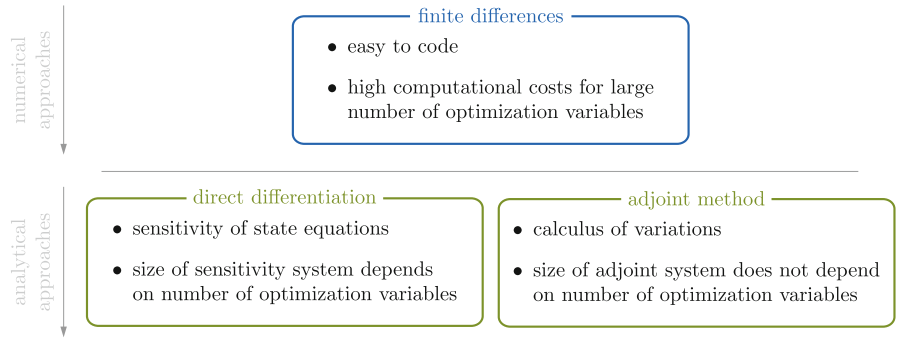

numerical approaches
finite differences
• easy to code
• high computational costs for large number of optimization variables
analytical approaches
direct differentiation
• sensitivity of state equations
• size of sensitivity system depends on number of optimization variables
adjoint method
• calculus of variations
• size of adjoint system does not depend on number of optimization variables

Fig. 2. Overview of approaches to compute first-order derivatives

in which the partial derivative of the parameterized control with respect to the grid nodes

$$\frac{\partial \mathbf{u}}{\partial \bar{\mathbf{u}}} = \mathbf{C} \tag{7}$$

has been utilized. Partial derivatives of an arbitrary function f with respect to x are denoted with subscripts, i.e. f_x. Similar to the gradient of the cost functional, the gradient of the final constraints in Eq. (3) can be calculated by direct differentiation as

$$\nabla_{\bar{\mathbf{u}}} \boldsymbol{\phi}^\mathsf{T} = \boldsymbol{\phi}_\mathbf{x} \mathbf{x}_{\bar{\mathbf{u}}}. \tag{8}$$

The resulting gradients in Eq. (6) and Eq. (8) involve the system sensitivity $\mathbf{x}_{\bar{\mathbf{u}}} \in \mathbb{R}^{n \times m \cdot k}$ which is obtained by differentiating the state equations with respect to the grid nodes as

$$\begin{align} \dot{\mathbf{x}}_{\bar{\mathbf{u}}} &= \mathbf{f}_\mathbf{x} \mathbf{x}_{\bar{\mathbf{u}}} + \mathbf{f}_\mathbf{u} \mathbf{u}_{\bar{\mathbf{u}}} \tag{9} \\ &= \mathbf{f}_\mathbf{x} \mathbf{x}_{\bar{\mathbf{u}}} + \mathbf{f}_\mathbf{u} \mathbf{C}. \tag{10} \end{align}$$

Initial conditions of the system sensitivity are defined as

$$\mathbf{x}_{\bar{\mathbf{u}}}(0) = \mathbf{0}, \tag{11}$$

since the initial conditions of the state equations do not depend on the grid nodes, i.e. $\mathbf{x}(0) = \mathbf{x}_0$. The system Jacobian $\mathbf{f}_\mathbf{x} \in \mathbb{R}^{n \times n}$ and $\mathbf{f}_\mathbf{u} \in \mathbb{R}^{n \times m}$ have to be calculated *a priori*, e.g., by analytical differentiation, in order to solve the matrix differential system in Eq. (10). Remark that the differential system depends on the number of grid nodes. Thus, the computational effort increases with the number of grid nodes.

3.2 Adjoint Gradient Approach for Discrete Control Parameterization

A large number of grid nodes leads to a large solution space and, therefore, the gradient computation leads to a high computational effort resulting from finite differences or direct differentiation. An efficient alternative to compute gradients analytically is the adjoint variable method which is based on the calculus of variations. Following the basic idea presented in the seventies by Bryson and Ho [1], an adjoint gradient approach for discrete control parameterizations is utilized. Lichtenecker et al. [6] derived the adjoint gradients for time-optimal control problems defined in Eqs. (1)–(3) for spline control parameterizations in the following form:

$$\nabla_{\bar{\mathbf{u}}} J^\mathsf{T} = \int_{t_0}^{t_f} \left(\mathbf{p}^\mathsf{T} \mathbf{f}_\mathbf{u} + P_\mathbf{u} \right) \mathbf{C} \, \mathrm{d}t, \tag{12}$$

$$\nabla_{\bar{\mathbf{u}}} \boldsymbol{\phi}^\mathsf{T} = \int_{t_0}^{t_f} \mathbf{R}^\mathsf{T} \mathbf{f}_\mathbf{u} \mathbf{C} \, \mathrm{d}t, \tag{13}$$

in which the adjoint variables fulfill the (adjoint) system of differential equations

$$\dot{\mathbf{p}} = -P_{\mathbf{x}}^{\mathsf{T}} - \mathbf{f}_{\mathbf{x}}^{\mathsf{T}}\mathbf{p} \qquad \text{with} \qquad \mathbf{p}(t_f) = \mathbf{0}, \tag{14}$$

$$\dot{\mathbf{R}} = -\mathbf{f}_{\mathbf{x}}^{\mathsf{T}}\mathbf{R} \qquad \text{with} \qquad \mathbf{R}(t_f) = \boldsymbol{\phi}_{\mathbf{x}}^{\mathsf{T}}(\mathbf{x}(t_f), t_f). \tag{15}$$

Due to the final conditions, they have to be solved backward in time to compute the adjoint gradients. Moreover, it has to be emphasized that the size of the adjoint system does not grow with the number of grid nodes which is not the case for direct differentiation, see Sect. 3.3. The adjoint gradients in Eqs. (12) and (13) prove to be preferable regarding computational effort and accuracy in gradient based optimization strategies. For further details on adjoint gradients, the reader is referred to [2,6].

How to Compute the Adjoint Gradients
The adjoint gradients in Eqs. (12) and (13) can be used for direct and indirect optimization algorithms. Both approaches are iterative methods and, therefore, the gradients have to be recomputed in each iteration. In this paper, we use a direct optimization method in order to compute the optimal control. Similar as shown in [10], Fig. 3 illustrates the application of adjoint gradients provided to a direct optimization method and is summarized with the following steps:

1. Select a direct optimization method which is able to use user-defined gradients, e.g. a classical SQP method or an Interior Point (IP) method.
2. The optimization algorithm proposes values $\mathbf{z}_i$ for the optimization variables associated to the current i-th iteration. Starting from this view, the gradients have to be computed for the $(i+1)$-th iteration.
3. Solve the state equations related to the actual optimization variables and initial conditions using an ODE solver.
4. The cost functional and the final constraints can be evaluated.
5. Compute the adjoint variables $\mathbf{p}$ and $\mathbf{R}$ backward in time using Eqs. (14) and (15).
6. Finally, the adjoint gradients of the cost functional and the final constraints are computed by a time integration and provided to the optimization algorithm for the next iteration.
7. Steps (2) through (6) are repeated until the KKT conditions are fulfilled with respect to the optimal solution $\mathbf{z}^*$.

3.3 Discussion on Duality of Gradients

McNamara et al. [7] pointed out that the adjoint approach can be interpreted as a special case of linear duality and that the core of this method is based on a substitution of variables. This can be seen by considering the first term of the gradients of the cost functional in Eqs. (6) and (13), i.e.,

$$\int_{t_0}^{t_f} P_{\mathbf{x}}\mathbf{x}_{\bar{\mathbf{u}}}\,\mathrm{d}t \quad \text{with} \quad \dot{\mathbf{x}}_{\bar{\mathbf{u}}} = \mathbf{f}_{\mathbf{x}}\mathbf{x}_{\bar{\mathbf{u}}} + \mathbf{f}_{\mathbf{u}}\mathbf{C} \quad \text{and} \quad \mathbf{x}_{\bar{\mathbf{u}}}(0) = \mathbf{0}, \tag{a}$$

$$\int_{t_0}^{t_f} \mathbf{p}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\mathbf{C}\,\mathrm{d}t \quad \text{with} \quad \dot{\mathbf{p}} = -P_{\mathbf{x}}^{\mathsf{T}} - \mathbf{f}_{\mathbf{x}}^{\mathsf{T}}\mathbf{p} \quad \text{and} \quad \mathbf{p}(t_f) = \mathbf{0}. \tag{b}$$

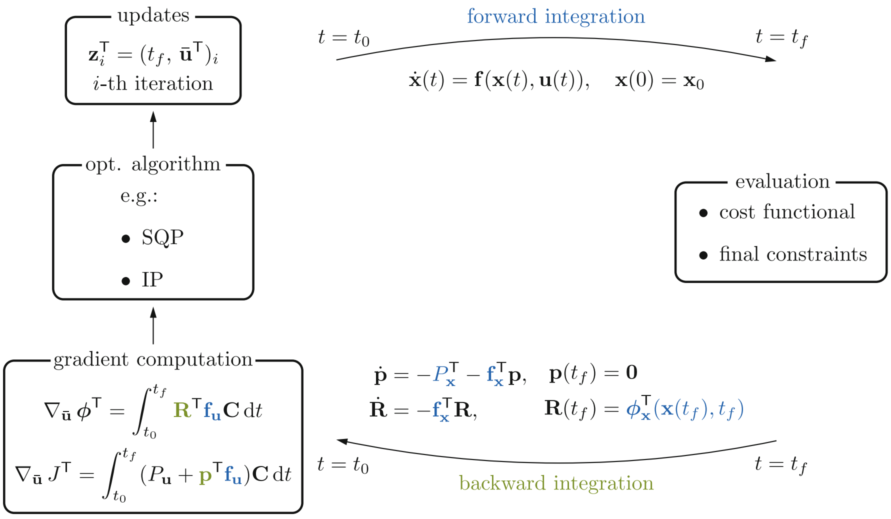

updates
$\mathbf{z}_i^\mathsf{T} = (t_f,\, \bar{\mathbf{u}}^\mathsf{T})_i$
i-th iteration
opt. algorithm
e.g.:
• SQP
• IP
gradient computation
$\nabla_{\bar{\mathbf{u}}}\,\phi^\mathsf{T} = \int_{t_0}^{t_f} \mathbf{R}^\mathsf{T}\mathbf{f}_\mathbf{u}\mathbf{C}\,\mathrm{d}t$
$\nabla_{\bar{\mathbf{u}}}\,J^\mathsf{T} = \int_{t_0}^{t_f} (P_\mathbf{u} + \mathbf{p}^\mathsf{T}\mathbf{f}_\mathbf{u})\mathbf{C}\,\mathrm{d}t$
forward integration
$t = t_0$
$t = t_f$
$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \quad \mathbf{x}(0) = \mathbf{x}_0$
evaluation
• cost functional
• final constraints
$\dot{\mathbf{p}} = -P_\mathbf{x}^\mathsf{T} - \mathbf{f}_\mathbf{x}^\mathsf{T}\mathbf{p}, \quad \mathbf{p}(t_f) = \mathbf{0}$
$\dot{\mathbf{R}} = -\mathbf{f}_\mathbf{x}^\mathsf{T}\mathbf{R}, \quad \mathbf{R}(t_f) = \phi_\mathbf{x}^\mathsf{T}(\mathbf{x}(t_f), t_f)$
$t = t_0$
$t = t_f$
backward integration

Fig. 3. Procedure for the use of adjoint gradients in direct optimization approaches

Both terms require the solution of a linear differential system, but it has to be emphasized that the size of the systems is different. The size of the system sensitivity depends on the number of states n, the number of controls m and on the number of grid nodes k, while the size of the adjoint system depends only on n. To compute the gradients, one can solve either the primal system (a) with dimension $(n \times m \cdot k)$ or the dual system (b) with dimension $(n \times 1)$. Thus, the adjoint approach is an efficient technique to incorporate especially a large number of grid nodes. A graphical interpretation of the dimensions occurring in the gradients of the cost functional is shown in Fig. 4, with a special focus on increasing the number of grid nodes.

4 Numerical Example

4.1 Task Description and Optimization Problem

The analytically derived adjoint gradients in [6] are used for a direct optimization method in a time-optimal control problem of a SCARA with two rigid bodies. The goal is to manipulate the tool center point (TCP) of the robot depicted in Fig. 5 from an initial state to a final state in minimal operation time t_f^* with a discrete control parameterization. To meet industrial requirements, the control is forced to be $\mathcal{C}^2$ continuous. Hence, the matrix $\mathbf{C}$ is chosen such that the interpolation of each discretized control subinterval is performed by a cubic spline function. The state equations are obtained by introducing the state variables

$$\mathbf{x} = (\varphi_1,\, \varphi_2,\, \omega_1,\, \omega_2)^\mathsf{T}, \tag{16}$$

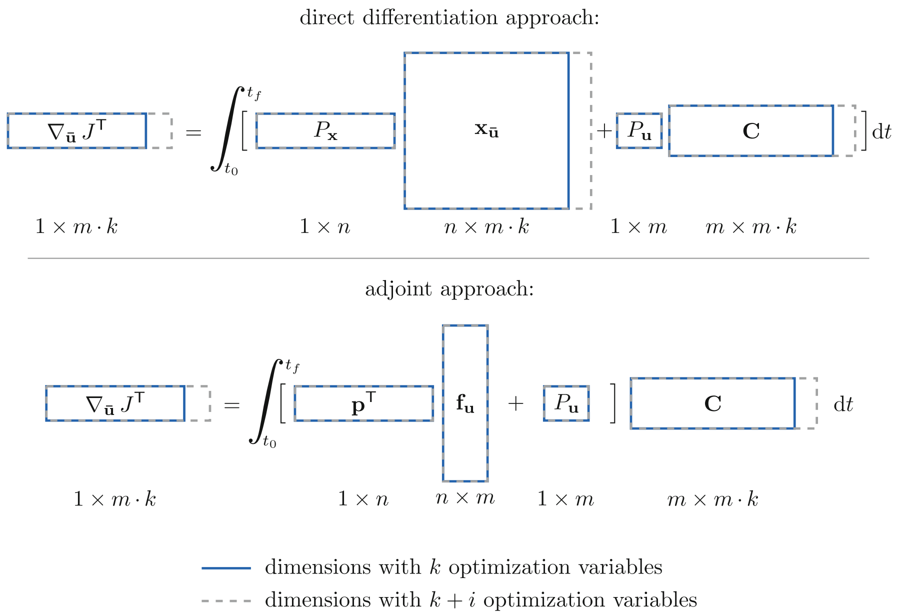


Fig. 4. Graphical interpretation of the dimensions occurring in the gradient of the cost functional with respect to the direct differentiation approach in Eq. (6) and the adjoint approach in Eq. (13)

in which $\dot{\varphi}_i = \omega_i$. The model parameters for the simulation are set as follows: $m_1 = m_3 = 1\,\mathrm{kg}$, $m_2 = 0.5\,\mathrm{kg}$, $l_i = 1\,\mathrm{m}$ and $J_i = m_i l_i^2/12$, in which $i \in \{1, 2\}$. The mass m_3 is considered as a point mass attached to the TCP.

The cost functional of the optimization problem is given in Eq. (2), in which the penalty term $P(\mathbf{u}) = 10(P_1(u_1) + P_2(u_2))$ is used with

$$P_i(u_i) := \begin{cases} 0 & \text{for } |u_i| < u_{i,\max}, \\ \frac{1}{2}(|u_i| - u_{i,\max})^2 & \text{otherwise.} \end{cases} \tag{17}$$

The final constraints of the system are defined as

$$\boldsymbol{\phi}(\varphi_1, \varphi_2, \omega_1, \omega_2) := \left. \begin{pmatrix} l_1 \cos(\varphi_1) + l_2 \cos(\varphi_2) - x_f \\ l_1 \sin(\varphi_1) + l_2 \sin(\varphi_2) - y_f \\ \omega_1 \\ \omega_2 \end{pmatrix} \right|_{t=t_f}, \tag{18}$$

in which $x_f = 1\,\mathrm{m}$ and $y_f = 1\,\mathrm{m}$ denote the desired final configuration of the TCP. Physical bounds of the controls are given by $u_{1,\max} = 4\,\mathrm{Nm}$ and $u_{2,\max} = 2\,\mathrm{Nm}$.

The NLP contains the optimization variables $\mathbf{z}^\mathsf{T} = (t_f, \bar{\mathbf{u}}^\mathsf{T})$ and is solved with an SQP method. As an initial guess, the assumption for the final time is $t_f = 2\,\mathrm{s}$ and the grid nodes are set to $\bar{\mathbf{u}} = \mathbf{0}$. Initial conditions of the state

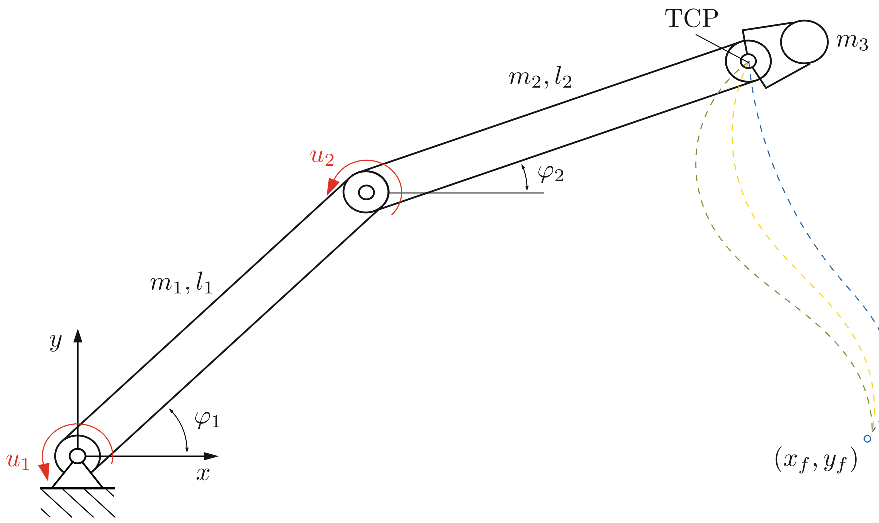


Fig. 5. SCARA with two rigid bodies in a general configuration

variables are set to $\mathbf{x}_0 = (-\pi/4,\ 0,\ 0,\ 0)^\mathsf{T}$. In order to analyze the sensitivity of the solution to the refinement of the discretization of the control, both controls are equidistantly discretized in the time interval $t \in [0, t_f]$ with a set of grid nodes with various number $k \in \{5, 10, 20, 30, 40, 50\}$.

4.2 Results

Figure 6 shows the optimal control history $\mathbf{u}_k^*$ and the resulting trajectory of the TCP with respect to the defined number of grid nodes k. One can observe that the control becomes a bang-bang type control by increasing the number of grid nodes. It can also be seen that the TCP trajectory with $k = 5$ grid nodes is noticeably different compared to controls in which the number of grid nodes is higher. This is due to the fact that in this case the optimal control cannot be represent a bang-bang structure. Theoretically, an infinite number of grid nodes will lead to the shortest possible final time. The final times for the six independent optimizations are $(k = 5, t_f^* = 1.9439\,\mathrm{s})$, $(10, 1.8633\,\mathrm{s})$, $(20, 1.8391\,\mathrm{s})$, $(30, 1.8325\,\mathrm{s})$, $(40, 1.8303\,\mathrm{s})$ and $(50, 1.8294\,\mathrm{s})$.

The optimal control with $k = 50$ grid nodes and the corresponding switching functions, as defined in [2] for bang-bang controls, are shown in Fig. 7. The zero values of the control agree well with those of the switching functions h_i and the Hamiltonian of the system is sufficiently small. Thus, the termination criteria shown in Fig. 1 is satisfied and a bang-bang control can be approximated with a sufficient number of grid nodes.

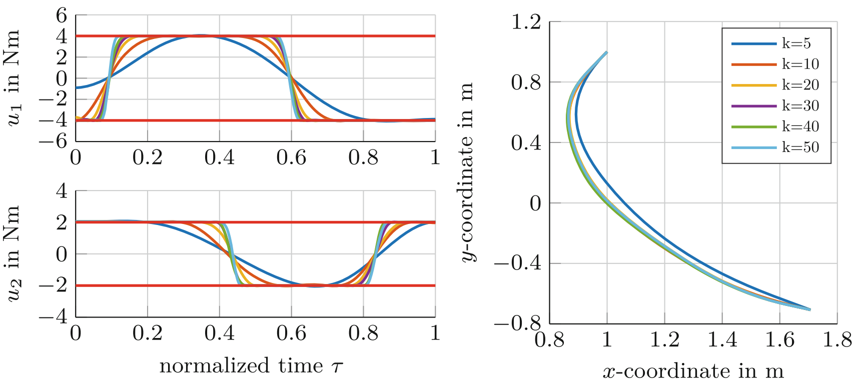


Fig. 6. Optimal control history and TCP trajectory for various number of grid nodes

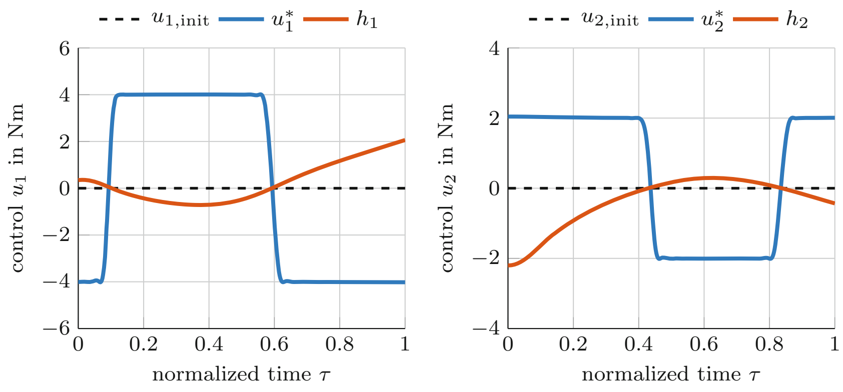


Fig. 7. Initial controls, optimal controls and switching functions considering a cubic spline parameterization of the control

5 Conclusions

This paper presents a procedure for using adjoint variables in a direct optimization approach. The adjoint variables are examined in the context of two scenarios: The adjoint variables are used to compute the gradients during the optimization. In addition, the adjoint variables are used to evaluate Pontryagin's minimum principle in order to discuss the optimization results obtained by an SQP method. A time-optimal control problem of a SCARA shows the versatile application of adjoint variables. Moreover, the computational effort for the computation of gradients can be reduced by considering adjoint gradients, especially when the number of grid nodes is large or the mechanical system is difficult to solve forward in time.

Acknowledgements. Daniel Lichtenecker and Karin Nachbagauer acknowledge support from the Technical University of Munich - Institute for Advanced Study.

References

1. Bryson, A.E., Ho, Y.C.: Applied Optimal Control: Optimization, Estimation, and Control. Taylor & Francis, New York (1975). https://doi.org/10.1201/9781315137667
2. Eichmeir, P., Nachbagauer, K., Lauß, T., Sherif, K., Steiner, W.: Time-optimal control of dynamic systems regarding final constraints. J. Comput. Nonlinear Dynam. **16**(3), 031003 (2021). https://doi.org/10.1115/1.4049334
3. Karush, W.: Minima of functions of several variables with inequalities as side constraints. Master's thesis, Department of Mathematics, University of Chicago (1939)
4. Kelley, H.J.: Method of gradients: optimization techniques with applications to aerospace systems. Math. Sci. Eng. **5**, 205–254 (1962)
5. Kuhn, H.W., Tucker, A.W.: Nonlinear programming. In: Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, pp. 481–492 (1951)
6. Lichtenecker, D., Rixen, D., Eichmeir, P., Nachbagauer, K.: On the use of adjoint gradients for time-optimal control problems regarding a discrete control parameterization. Multibody Sys. Dyn. **59**(3), 313–334 (2023). https://doi.org/10.1007/s11044-023-09898-5
7. McNamara, A., Treuille, A., Popović, Z., Stam, J.: Fluid control using the adjoint method. ACM Trans. Graph. **23**(3), 449–456 (2004). https://doi.org/10.1145/1015706.1015744
8. Nachbagauer, K., Oberpeilsteiner, S., Sherif, K., Steiner, W.: The use of the adjoint method for solving typical optimization problems in multibody dynamics. J. Comput. Nonlinear Dynam. **10**(6), 061011 (2015). https://doi.org/10.1115/1.4028417
9. Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd edn. Springer, New York (2006). https://doi.org/10.1007/978-0-387-40065-5
10. Pikuliński, M., Malczyk, P.: Adjoint method for optimal control of multibody systems in the Hamiltonian setting. Mech. Mach. Theory **166**, 104473 (2021). https://doi.org/10.1016/j.mechmachtheory.2021.104473
11. Pontryagin, L.S., Boltyanskii, V.G., Gamkrelidze, R.V., Mischchenko, E.F.: The Mathematical Theory of Optimal Processes. John Wiley & Sons, New York (1962)
12. Reiter, A., Müller, A., Gattringer, H.: On higher order inverse kinematics methods in time-optimal trajectory planning for kinematically redundant manipulators. IEEE Trans. Ind. Inf. **14**(4), 1681–1690 (2018). https://doi.org/10.1109/TII.2018.2792002

Author Index