robotics

IMPACT FACTOR 2.9

CITESCORE 6.7

Article

A Numerical Integrator for Kinetostatic Folding of Protein Molecules Modeled as Robots with Hyper Degrees of Freedom

Amal Kacem, Khalil Zbiss and Alireza Mohammadi

Special Issue

Bioinspired Robotics: Toward Softer, Smarter and Safer

Edited by

Prof. Dr. Adrian Burlacu

MDPI

https://doi.org/10.3390/robotics13100150

robotics

MDPI

Article

A Numerical Integrator for Kinetostatic Folding of Protein Molecules Modeled as Robots with Hyper Degrees of Freedom

Amal Kacem †, Khalil Zbiss † and Alireza Mohammadi *

Department of Electrical & Computer Engineering, College of Engineering & Computer Science, University of Michigan–Dearborn, 2057 Tony England ELB, Dearborn, MI 48128, USA; akacem@umich.edu (A.K.); kzbiss@umich.edu (K.Z.)

* Correspondence: amohmmad@umich.edu

† These authors contributed equally to this work.

Abstract: The kinetostatic compliance method (KCM) models protein molecules as nanomechanisms consisting of numerous rigid peptide plane linkages. These linkages articulate with respect to each other through changes in the molecule dihedral angles, resulting in a kinematic mechanism with hyper degrees of freedom. Within the KCM framework, nonlinear interatomic forces drive protein folding by guiding the molecule's dihedral angle vector towards its lowest energy state in a kinetostatic manner. This paper proposes a numerical integrator that is well suited to KCM-based protein folding and overcomes the limitations of traditional explicit Euler methods with fixed step size. Our proposed integration scheme is based on pseudo-transient continuation with an adaptive step size updating rule that can efficiently compute protein folding pathways, namely, the transient three-dimensional configurations of protein molecules during folding. Numerical simulations utilizing the KCM approach on protein backbones confirm the effectiveness of the proposed integrator.

Keywords: kinetostatic compliance method; nanomechanisms; kinetostatic robot models; protein folding; numerical integrators; hyper degrees of freedom

check for updates

Citation: Kacem, A.; Zbiss, K.; Mohammadi, A. A Numerical Integrator for Kinetostatic Folding of Protein Molecules Modeled as Robots with Hyper Degrees of Freedom. *Robotics* **2024**, *13*, 150. https://doi.org/10.3390/robotics13100150

Academic Editor: Seokheun Choi

Received: 12 August 2024
Revised: 27 September 2024
Accepted: 30 September 2024
Published: 2 October 2024

1. Introduction

Kinetostatic modeling is a cornerstone in the design, analysis, motion control, and optimization of robotic systems [1]. This approach has proven effective in such diverse robotics applications as design of underactuated origami grippers [2], tendon-driven parallel continuum robots [3], and concentric push–pull robots [4].

Kinetostatic modeling has also proven remarkably effective in determining the transient and final three-dimensional structures of proteins during folding [5–7]. The kinetostatic approach to protein folding leverages the fundamental concept that *proteins can be represented as nanomechanisms consisting of rigid peptide plane linkages* (see, e.g., [7–13]). These linkages form a kinematic chain with hyper degrees of freedom; for instance, even a very small synthetic protein such as chignolin has nine peptide planes, corresponding to eighteen degrees of freedom in the backbone chain [14]. This allows them to articulate with respect to each other through rotations in dihedral angles centered at alpha-Carbon atoms, as illustrated in Figure 1 (see Section 2.1 for further details). Hence, it should be no surprise that techniques from robotic motion planning [11,15,16] and control theory [12,13,17–19] have been utilized to investigate the problem of protein folding/unfolding.

When modeling proteins as kinematic nanomechanisms, the kinetostatic compliance method (KCM) views protein folding as a process where the molecular configuration changes due to electrostatic and van der Waals nonlinear interatomic forces [20]. In this model, the peptide linkages articulate with respect to each other through dihedral angle rotations. Since its introduction, the KCM has proven to be a valuable tool for understanding hydrogen bond formation and its impact on protein mobility [21]. Furthermore, it has enabled the design of peptide-based nanorobots and nanomachines [22–25]. Indeed,

KCM-based folding simulations can provide quantitative prediction of mobility and range of motion in protein-based nanomachines such as parallel nanorobotic platforms [26].

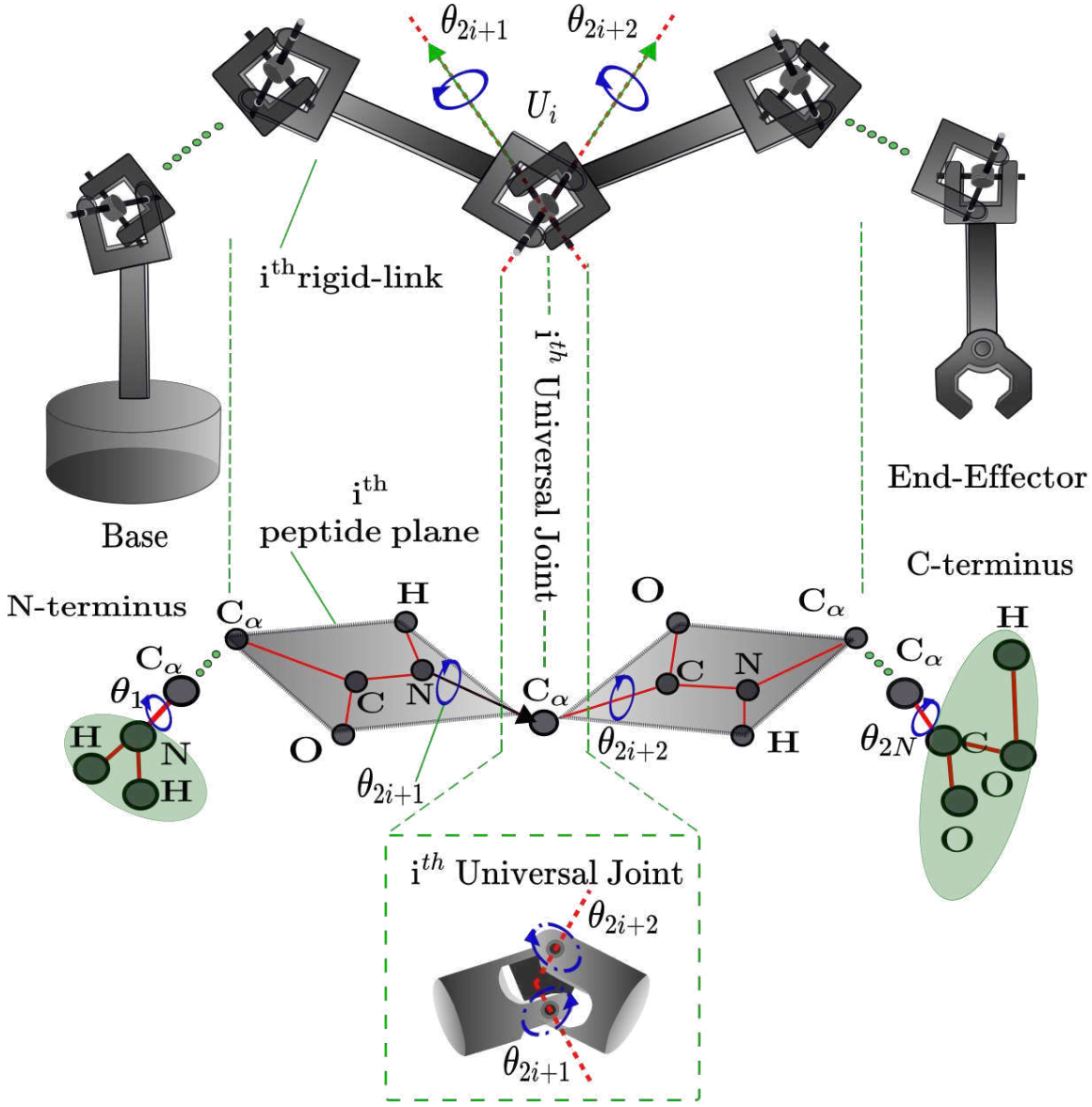

Figure 1. The kinematic structure of the protein backbone chain is similar to that of robotic mechanisms with hyper degrees of freedom. Specifically, C_α atoms play the role of hinges connecting peptide planes together. These C_α atoms are kinematically the same as universal joints with two degrees of freedom. In kinetostatic protein folding, the peptide linkages articulate with respect to each other through dihedral angular variations facilitated by the C_α atoms.

To predict and control robotic system trajectories, roboticists typically employ dynamic models expressed as ordinary or partial differential equations. Therefore, devising an appropriate integration scheme considering a robot's specific equations of motion and application is crucial. Table 1 provides a sample overview of numerical integration algorithms used in the robotics literature, ranging from implicit Euler methods for tendon-driven robots to Lie group variational integrators for underwater autonomous vehicles.

Table 1. Sample overview of numerical integration schemes utilized for diverse robotic systems/applications.

Reference	Numerical Integrator	Application Area
[27] Testard et al. (2023)	Implicit Euler integrator	Elastic tendon-driven robots
[28] Gibson & Murphey (2010)	Variational integrator	Orbital docking of Canadarm and the ISS
[29] Pekarek & Marsden (2008)	Variational collision integrator	Legged robotic locomotion
[30] Fan et al. (2018)	Higher-order variational integrators	Robot trajectory optimization
[31] Braun & Goldfarb (2012)	Explicit DAE integrator	Constrained mechanical systems
[32] Fang et al. (2023)	Half-implicit integrator	Index three DAEs in multibody dynamics
[33] Till et al. (2019)	Runge–Kutta (RK4) integrator	Continuum and soft robots
[34] Nordvik & Sanyal (2010)	Lie group variational integrator	Underwater autonomous vehicles
[12] Mohammadi & Spong (2022)	Explicit Euler integrator	Protein molecules modeled as nano-mechanisms with hyper degrees of freedom

While the KCM approach offers computational advantages for simulating kinetostatic protein folding, its reliance on explicit Euler integration with a fixed step size hinders its efficiency. Indeed, each integration step necessitates intensive calculations of interatomic forces, incorporation of physical constraints, and conversion to kinetostatic input torques. Consequently, the explicit Euler method's convergence to a folded conformation is often computationally expensive due to the excessive number of required iterations. To overcome these limitations, a fast and stable numerical algorithm is essential for accelerating large-scale KCM-based protein folding simulations.

In this paper, we develop an effective numerical integration technique for KCM-based protein folding. Our proposed integrator relies on the explicit pseudo-transient continuation framework (ΨTC, our choice of ΨTC for abbreviating "pseudo-transient continuation", follows the pioneering work of Kelley, Coffey, and collaborators; see, e.g., [35,36]). ΨTC is an efficient numerical method for calculating steady-state solutions of systems arising from ODE- or PDE-based transient dynamics [35–40]. The technique adapts the step sizes base on the underlying dynamics and proximity to the steady state. These updates are rooted in variations of the *switched evolution relaxation* (SER) method by Mulder and van Leer [41]. Thanks to its robust convergence and stability properties, ΨTC has been applied in diverse fields such as fluid dynamics [42], radiation transport [43], and magnetohydrodynamics [44].

Contributions of the Paper: The present paper advances the KCM-based protein folding framework by introducing an explicit ΨTC numerical integrator equipped with adaptive step-size control tailored to kinetostatic protein folding. Unlike previous KCM studies that exclusively relied on explicit Euler integrators with a fixed step size, our approach offers enhanced efficiency and accuracy. We provide rigorous analysis of the numerical stability and convergence properties of the explicit ΨTC integrator within the kinetostatic protein folding context. Leveraging these properties, our method converges to folded protein conformations with fewer computational steps than traditional KCM approaches. Hence, the proposed method results in a significant reduction of the computational cost associated with KCM-based folding numerical simulations. Furthermore, our proposed numerical integration technique has potential beyond kinetostatic protein folding. Leveraging the similarity between the protein kinematic structure in the KCM framework and robotic manipulators with hyper degrees of freedom [45], our method is also applicable to multi-section continuum robots [46,47]. Moreover, its potential for fast numerical integration makes it a strong candidate for real-time implementation of model predictive controllers in soft robotic arms [48].

This paper significantly expands upon a preliminary version presented as a conference poster [49]. In this paper, we delve deeper into the ΨTC numerical integration algorithm by providing a detailed description and discussing its numerical properties in the context of kinetostatic protein folding. Additionally, we present a detailed analysis (absent from the

conference poster) investigating the stability and convergence properties of the proposed ΨTC numerical integrator. Finally, the algorithm effectiveness is rigorously assessed through more extensive numerical simulations on protein backbone chains with many degrees of freedom.

The rest of this paper is organized as follows. First, in Section 2 we provide an overview of the kinematics of protein molecules and the KCM framework for modeling the protein folding process. In Section 3, we present the problem statement and elaborate the computational needs of kinetostatic protein folding. Next, in Section 4 we present our explicit ΨTC integration scheme and an adaptive step-size control strategy for the underlying kinetostatic protein folding process. After presenting the numerical simulation results in Section 5, we conclude the paper with future research directions and final remarks in Section 6.

Notation: We denote the set of all non-negative real numbers by $\mathbb{R}_+$. Given a vector $\mathbf{x}$ in $\mathbb{R}^M$ and a real constant $p \geq 1$, we denote its p-norm by $|\mathbf{x}|_p$. Given an integer M and matrix $\mathbf{A}$ in $\mathbb{R}^{M\times M}$, we let $\varrho(\mathbf{A})$ denote the spectral radius of $\mathbf{A}$. We say that a sequence $\{\mathbf{x}_k\}_{k=1}^{\infty} \subset \mathbb{R}^M$ *converges q-linearly* to $\mathbf{L}$ if there exists $\kappa \in (0,1)$ such that $\lim\limits_{k\to\infty} \frac{|\mathbf{x}_{k+1}-\mathbf{L}|}{|\mathbf{x}_k-\mathbf{L}|} = \kappa$.

2. Background

In this section we provide a summary of the KCM framework for modeling the kinetostatic folding of protein molecules in vacuo. To prevent ambiguity, we note that "conformation" is the established term in the field of biochemistry for denoting the spatial arrangement of a protein molecule's kinematic structure; conversely, the field of robot kinematics predominantly employs "configuration" for the same concept. In this paper, we utilize "conformation" and "configuration" interchangeably.

2.1. Nano-Linkage-Based Kinematic Model of Protein Molecules

Proteins are macromolecules with complex structures and intricate dynamics. These long molecular chains consist of peptide planes that are interconnected by chemical bonds. The functionality of protein molecules depends critically on their three-dimensional (3D) structure (i.e., their conformation), which can be directly deduced from the linear sequence of amino acids forming their polypeptide chain. For brevity, the following presentation is confined to the protein backbone.

A schematic of the protein polypeptide backbone chain is depicted in Figures 1 and 2. As can be seen from the figure, each individual peptide plane is formed by six coplanar atoms that are covalently bonded (visualized as red lines in Figure 2). The coplanarity assumption about the α-carbon C_α, carbonyl CO, amide nitrogen NH, and another α-carbon is motivated by the wealth of related high-resolution X-ray crystallographic data (see, e.g., [50]). This fundamental assumption underpins the conceptualization of protein molecules as intricate nanomechanisms with numerous degrees of freedom (see, e.g., [5,8,12,13,51]). *Accordingly, the key building blocks of the protein kinematic chain are the rigid peptide planes, which function as rigid nanoscale linkages (see Figures 1 and 2).*

As illustrated in Figures 1 and 2, the rotation of the nano-linkages in the protein kinematic chain with respect to each other is facilitated by the central carbon atoms, which are denoted by C_α and commonly known as the alpha-Carbon atoms. Indeed, C_α atoms play the role of hinges connecting peptide planes together; C_α carbon forms chemical bonds with four distinct constituents: the nitrogen, hydrogen, and carbonyl carbon atoms of the peptide bond and a variable side chain represented by SR. The first and last C_α atoms are bonded to an N-terminus (an amino group)–peptide plane pair and a C-terminus (a carboxyl group)–peptide plane pair, respectively.

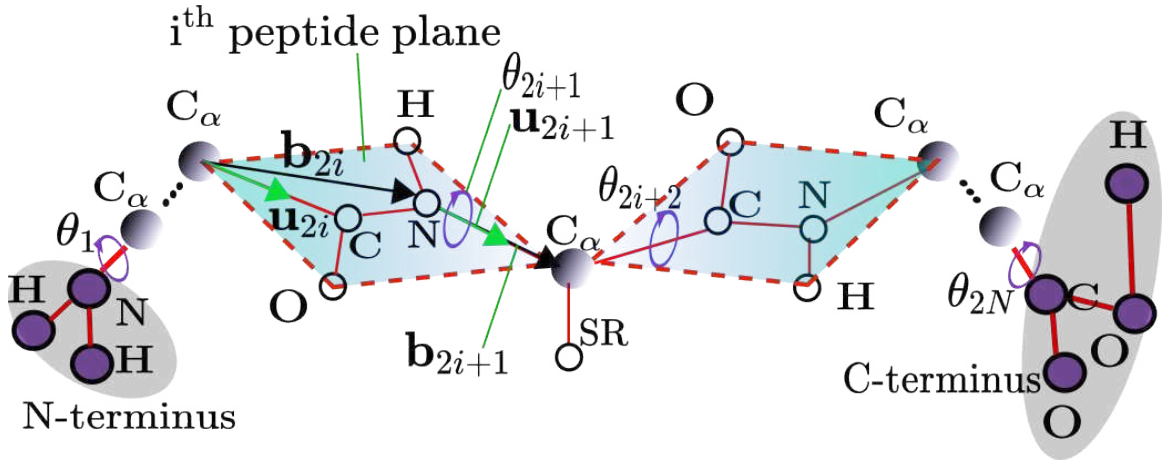

Figure 2. Modeling protein backbone chains as nano-kinematic structures in which peptide planes act as rigid links interconnected by revolute joints centered at the alpha carbon atoms, each offering two degrees of freedom. The nano-kinematic chain can rotate about the unit vectors $\mathbf{u}_j$, $1 \leq j \leq 2N$, which align with the $C_\alpha - C$ and $N - C_\alpha$ bonds. The body vectors, denoted by $\mathbf{b}_j$ and $1 \leq j \leq 2N$, comprehensively define the relative spatial arrangement of coplanar atoms within each peptide plane. The body and unit vectors act in concert to provide a comprehensive description of the protein molecule's 3D configuration as the dihedral angles θ_j and $1 \leq j \leq 2N$ rotate under the influence of interatomic forces.

Protein Configuration Vector: The nano-kinematic structure of a protein molecule comprising $N-1$ peptide planes can be comprehensively represented by a set of bond lengths and dihedral angle pairs. Considering the i^{th} peptide plane $1 \leq i \leq N-1$, each pair of these dihedral angles represent rotations about the covalent bonds $N - C_\alpha$, namely, θ_{2i+1}, and $C_\alpha - C$, i.e., θ_{2i+2} (Figure 2). Additionally, the dihedral angle vector

$$\boldsymbol{\theta} := \left[\theta_1, \cdots, \theta_{2N}\right]^\top \tag{1}$$

represents the configuration of the backbone chain. Therefore, the vector $\boldsymbol{\theta}$ encompassing the dihedral angles resides within the $2N$-dimensional configuration space $\mathcal{Q}$. This space, formally defined as $\mathcal{Q} := \mathcal{S}^1 \times \cdots \times \mathcal{S}^1$, represents the Cartesian product of $2N$ individual unit circles $\mathcal{S}^1$, each corresponding to a single dihedral angle within the backbone chain.

Each dihedral angle in the conformation vector $\boldsymbol{\theta}$ in Equation (1) corresponds to a single degree of freedom in the protein molecule's kinematic chain. For each degree of freedom, we can define a unit vector $\mathbf{u}_j$, $1 \leq j \leq 2N$. These vectors align with the axes of rotation around which the nano-kinematic chain can rotate. As illustrated in Figure 2, the unit vectors $\mathbf{u}_{2i}$ and $\mathbf{u}_{2i+1}$ represent the directions along two crucial bonds within the i^{th} peptide plane: the $C_\alpha - C$ bond and the $N - C_\alpha$ bond. These bonds define the primary rotation axes within the chain's segments. Serving as key structural references, $\mathbf{u}_1$ and $\mathbf{u}_{2N}$ denote the unit vectors associated with the amino terminus (N-terminus) and carboxyl terminus (C-terminus), respectively, of the polypeptide chain. These termini define the endpoints of the entire nano-kinematic chain. As illustrated in Figure 1, the N-terminus and C-terminus can be considered as the base and end-effector, respectively, of the protein nano-kinematic mechanism.

While dihedral angles define the rotational degrees of freedom within a protein's kinematic chain, additional information is necessary to fully capture the spatial orientation of the rigid peptide nano-linkages. This information comes in the form of body vectors, denoted by $\mathbf{b}_j$, $1 \leq j \leq 2N$. These vectors comprehensively define the relative spatial arrangement of coplanar atoms within each peptide plane. Consequently, any relative positional relationship between two atoms within a plane can be mathematically expressed as a linear combination of their corresponding body vectors, i.e., $k_{1m}\mathbf{b}_{2i} + k_{2m}\mathbf{b}_{2i+1}$, where the coefficients k_{1m} and k_{2m}, $1 \leq m \leq 4$ are constants specific to the atom pair and are

identical throughout the peptide chain (see, e.g., [5,6] for more details). The body vectors $\mathbf{b}_j$ and unit vectors $\mathbf{u}_j$ act in concert to provide a comprehensive depiction of the protein molecule's conformation as it dynamically varies with respect to the vector $\boldsymbol{\theta}$, which encompasses the dihedral angles.

Protein Forward Kinematics: The intricate interplay between the dihedral angle vector $\boldsymbol{\theta}$ and a protein's kinematic structure can be elegantly described through a framework of rotational matrix transformations [8,52]. This formalism offers a comprehensive and computationally efficient approach for representing protein conformation as a function of its internal torsional degrees of freedom. A reference configuration, designated by $\boldsymbol{\theta} = \mathbf{0}$ with reference unit vectors $\mathbf{u}_j^0$ and reference body vectors $\mathbf{b}_j^0$, $1 \leq j \leq 2N$, serves as the starting point for the following transformations:

$$\begin{aligned} \mathbf{u}_j(\boldsymbol{\theta}) &= \mathbf{X}(\boldsymbol{\theta}, \mathbf{u}_j^0)\mathbf{u}_j^0 \\ \mathbf{b}_j(\boldsymbol{\theta}) &= \mathbf{X}(\boldsymbol{\theta}, \mathbf{u}_j^0)\mathbf{b}_j^0 \end{aligned} \tag{2}$$

which compute the mapping from the dihedral angle space to the protein kinematic configuration. Each of the transformation matrices $\mathbf{X}(\boldsymbol{\theta}, \mathbf{u}_j^0)$, $1 \leq j \leq 2N$ in (2) is defined using sequential matrix multiplication:

$$\mathbf{X}(\boldsymbol{\theta}, \mathbf{u}_j^0) := \prod_{r=1}^{j} \mathbf{R}(\theta_j, \mathbf{u}_j^0) \tag{3}$$

where the rotation matrix $\mathbf{R}(\theta_j, \mathbf{u}_j^0)$ describes the rotation about the vector $\mathbf{u}_j^0$ with angle θ_j and the matrix $\Xi(\boldsymbol{\theta}, \mathbf{u}_j^0)$ determines the molecule kinematic structure using the dihedral angle conformation vector $\boldsymbol{\theta}$. In Equation (3), the rotation matrix $R(\theta_j, \mathbf{u}_j^0) \in \mathrm{SO}(3)$ describes the rotation about the direction provided by the unit vector $\mathbf{u}_j^0$ with an angle equal to θ_j. We remark that the special orthogonal group SO_3 is the set of all rotational matrices about the origin of three-dimensional Euclidean space; furthermore, any rotation matrix $R(\alpha, \hat{\mathbf{v}})$, where α is an angle and $\hat{\mathbf{v}} = [\hat{v}_x, \hat{v}_y, \hat{v}_z]^\top$ is a unit vector, can be written as

$$R(\alpha, \hat{\mathbf{v}}) = \begin{bmatrix} \hat{v}_x^2 V_\alpha + C_\alpha & \hat{v}_x \hat{v}_y V_\alpha - \hat{v}_z S_\alpha & \hat{v}_x \hat{v}_z V_\alpha + \hat{v}_y S_\alpha \\ \hat{v}_x \hat{v}_y V_\alpha + \hat{v}_z S_\alpha & \hat{v}_y^2 V_\alpha + C_\alpha & \hat{v}_y \hat{v}_z V_\alpha - \hat{v}_x S_\alpha \\ \hat{v}_x \hat{v}_z V_\alpha - \hat{v}_y S_\alpha & \hat{v}_y \hat{v}_z V_\alpha + \hat{v}_x S_\alpha & \hat{v}_z^2 V_\alpha + C_\alpha \end{bmatrix}, \tag{4}$$

where $V_\alpha := 1 - \cos(\alpha)$, $C_\alpha := \cos(\alpha)$, and $S_\alpha := \sin(\alpha)$.

Subsequent to the computation of body vectors $\mathbf{b}_j(\boldsymbol{\theta})$ (Equation (2)) and anchoring the N -terminal atom at the origin, the Cartesian coordinates of the k^{th}-peptide plane atoms can be determined by

$$\mathbf{r}_i(\boldsymbol{\theta}) = \sum_{j=1}^{i} \mathbf{b}_j(\boldsymbol{\theta}), \quad 1 \leq i \leq 2N - 1, \tag{5}$$

where the indices $i = 2k - 1$ and $i = 2k$ correspond to the nitrogen and C_α atoms, respectively.

2.2. Kinetostatic Compliance Folding

The KCM framework leverages the established experimental observation that it is possible to capture the essence of protein folding dynamics while neglecting inertial forces (see, e.g., [18,19,51,53]). According to the KCM, protein dihedral angles evolve kinetostatically under the influence of interatomic force fields.

Interatomic Force Fields Responsible for Kinetostatic Folding: Consider a peptide chain composed of N_a atoms and $N - 1$ peptide planes. The dihedral angle vector, denoted by $\boldsymbol{\theta}$ defined in Equation (1), encodes the conformational state of the chain. The Cartesian coordinates of any two atoms a_i, a_j within the chain are represented by $r_i(\boldsymbol{\theta})$, $r_j(\boldsymbol{\theta})$, respectively (Equation (5)). Their Euclidean distance, a fundamental metric for characterizing inter-

atomic interactions, is then calculated as $d_{ij}(\boldsymbol{\theta}) := |r_i(\boldsymbol{\theta}) - r_j(\boldsymbol{\theta})|_2$. The specific parameters pertaining to atom charges, van der Waals radii, interatomic distances, dielectric constant, potential well depths, and force weights are available in [6] and the provided references.

Within this framework, the total free energy responsible for protein folding, denoted by $\mathcal{G}(\boldsymbol{\theta})$, decomposes into electrostatic and van der Waals contributions as expressed in

$$\mathcal{G}(\boldsymbol{\theta}) := \mathcal{G}^{\text{elec}}(\boldsymbol{\theta}) + \mathcal{G}^{\text{vdw}}(\boldsymbol{\theta}), \tag{6}$$

where $\mathcal{G}^{\text{elec}}(\boldsymbol{\theta})$ and $\mathcal{G}^{\text{vdw}}(\boldsymbol{\theta})$ represent the protein's electrostatic potential energy and van der Waals interatomic potential energy, respectively (detailed expressions can be found in [5]). Consequently, the net forces acting on each atom a_i, $1 \leq i \leq N_{\text{a}}$ due to Coulombic and van der Waals interactions can be obtained through negative gradients of the respective individual energy terms: $F_i^{\text{elec}}(\boldsymbol{\theta}) = -\nabla_{\mathbf{r}_i}\mathcal{G}^{\text{elec}}$ and $F_i^{\text{vdw}}(\boldsymbol{\theta}) = -\nabla_{\mathbf{r}_i}\mathcal{G}^{\text{vdw}}$.

Kinetostatic Folding Torque Vector: The KCM-based modeling framework [5] necessitates calculation of the resultant forces and torques acting on each of the $N-1$ peptide planes within the protein molecule. These computed forces and torques are subsequently concatenated into a $6N$-dimensional vector $\mathcal{F}(\boldsymbol{\theta})$, which serves as the generalized force vector driving the protein folding process. However, in order to direct the actual changes in the protein's configuration, the vector $\mathcal{F}(\boldsymbol{\theta})$ needs to be mapped to an equivalent $2N$-dimensional torque vector, denoted as $\boldsymbol{\tau}(\boldsymbol{\theta})$. This mapping translates the generalized force vector into the specific torsional modifications exerted on the dihedral angles, ultimately governing the process of kinetostatic protein folding. The mathematical expression for the *kinetostatic folding torque vector* is expressed by

$$\boldsymbol{\tau}(\boldsymbol{\theta}) = \mathcal{J}^{\top}(\boldsymbol{\theta})\mathcal{F}(\boldsymbol{\theta}), \tag{7}$$

where $\mathcal{J}(\boldsymbol{\theta}) \in \mathbb{R}^{6N\times 2N}$ represents the molecule chain Jacobian at conformation $\boldsymbol{\theta}$. This matrix, acting as a bridge between generalized forces and torques, translates the $6N$-dimensional force vector into the corresponding $2N$-dimensional torque vector dictating the dihedral angle modifications (see [5] for detailed derivations). Consequently, the changes in dihedral angles at each protein conformation take place under the direct influence of the kinetostatic folding torque vector acting on the peptide backbone. Specifically, the vector $\boldsymbol{\tau}(\boldsymbol{\theta}) \in \mathbb{R}^{2N}$, which aligns with the steepest-descent direction of the total free energy $\mathcal{G}(\boldsymbol{\theta})$ in the protein conformation landscape, guides the dihedral angle variations during folding.

Folded Protein Conformations: Within the context of protein folding, at each local minimum $\boldsymbol{\theta}^*$ of the aggregate free energy function $\mathcal{G}(\boldsymbol{\theta})$ defined in Equation (6), the corresponding torque vector $\boldsymbol{\tau}(\boldsymbol{\theta}^*)$ vanishes identically; this signifies the protein molecule's kinetostatic stationarity at folded conformations, where the absence of net kinetostatic torque reflects the balanced internal forces within the protein structure. Equation (7) specifies the torque vector $\boldsymbol{\tau}(\boldsymbol{\theta})$ aligned with the steepest-descent direction of the free energy gradient in the conformational landscape of the protein molecule. As described below, Kazerounian and colleagues [5,8] in their pioneering KCM framework leveraged a normalized kinetostatic folding torque vector in an iterative manner.

Successive Kinetostatic Folding Iteration: Within the KCM framework, dihedral angles evolve kinetostatically under the influence of the kinetostatic folding torque vector resulting from the interatomic force fields. In particular, given an unfolded protein molecule conformation $\boldsymbol{\theta}_0$, the established kinetostatic compliance method relates joint torques to dihedral angle changes via the following numerical scheme (see, e.g., [5,51]):

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \kappa_0 \frac{\boldsymbol{\tau}(\boldsymbol{\theta}_k)}{|\boldsymbol{\tau}(\boldsymbol{\theta}_k)|_\infty}, \; k \in \mathbb{Z}_{0+} \tag{8}$$

where $\mathbb{Z}_{0+}$ denotes the set of non-negative integers, the ∞-norm of $\boldsymbol{\tau}(\boldsymbol{\theta}_k)$ is defined as $|\boldsymbol{\tau}(\boldsymbol{\theta}_k)|_\infty := \max_i |\boldsymbol{\tau}_i(\boldsymbol{\theta}_k)|$, and $\boldsymbol{\tau}(\boldsymbol{\theta}_k) = \mathcal{J}^{\top}(\boldsymbol{\theta}_k)\mathcal{F}(\boldsymbol{\theta}_k)$ according to Equation (7).

In the dihedral angle update iterations provided by (8), the normalized torque vector $\frac{\boldsymbol{\tau}(\boldsymbol{\theta}_k)}{|\boldsymbol{\tau}(\boldsymbol{\theta}_k)|_\infty}$ governs the incremental updates to the dihedral angles at each conformation $\boldsymbol{\theta}_k$.

As demonstrated by [12], the iteration provided by Equation (8) can be considered as an explicit Euler integration with time step κ_0. Crucially, the integration time step κ_0, which is determined through a heuristic procedure, has to be chosen sufficiently small to prevent extensive angle variations (see Section 2.3 for the resulting computational implications). Finally, we remark that the interaction between joint moments and dihedral angle variations in the kinetostatic folding process can theoretically lead to redundant calculations if not handled properly. However, our proposed method in this paper avoids this issue through the use of the explicit pseudo-transient continuation (ΨTC) integrator incorporating an adaptive step-size updating rule. This adaptive step size ensures that the step size increases as the system approaches a stable conformation, helping to avoid unnecessary iterations and redundant recalculations of joint moments. This also reduces the computational complexity, as the system requires fewer torque evaluations in regions of the energy landscape near equilibrium.

Convergence Criterion: The iterative process governed by Equation (8) continues until the molecule's aggregated free energy $\mathcal{G}(\boldsymbol{\theta})$ converges to the vicinity of a local minimum within the free energy landscape. Convergence is achieved numerically when the kinetostatic folding torque vector norm falls below a predefined tolerance $\tau_{\text{tol}} > 0$, i.e., when $|\boldsymbol{\tau}(\boldsymbol{\theta}_k)|_2 < \tau_{\text{tol}}$, where $|\boldsymbol{\tau}(\boldsymbol{\theta}_k)|_2$ denotes the Euclidean norm of $\boldsymbol{\tau}(\boldsymbol{\theta}_k)$.

2.3. Computational Burden of Kinetostatic Folding Iterations

A flowchart of the successive kinetostatic folding iteration is depicted in Figure 3. The *most computationally intensive* procedure at each conformation of the protein molecule (highlighted in red) consists of the electrostatic and van der Waals force computations. If exact interatomic force calculations are desired, then the computational complexity of this step for a molecule with N_a atoms is quadratic (i.e., of order $O(N_a^2)$) [5]. The computational complexity of numerical algorithms can be defined using big O notation, as follows: consider any two real-valued functions $h_1(\cdot)$ and $h_2(\cdot)$; if there exist a real number $a_0 > 0$ and real number z_0 such that the inequality $|h_1(z)| \leq a_0|h_2(z)|$ is satisfied for all $z \geq z_0$, then we say that $h_1(z) = O(h_2(z))$. Moreover, to encode more physical constraints in protein folding numerical simulations, such as entropy-loss constraints [12], it is necessary to solve a box-constrained convex quadratic program (QP) at each conformation of the protein molecule. The computational cost of such a convex QP using a state-of-the-art interior-point QP solver is of order $O(N^3)$ [54], where there are $N-1$ peptide planes in the protein backbone chain.

The variation of dihedral angles in the flowchart (the dashed border) needs to be governed by a proper kinetostatic folding update rule. For the conventional KCM iteration, the dihedral angle update rule governing the kinetostatic folding is provided by Equation (8). However, preventing numerical instability and large dihedral angle variations requires choosing sufficiently small values for the integration time step κ_0. This inevitably results in an excessive number of iterations for convergence to a folded protein molecule configuration. Consequently, *there is a need for departure from explicit Euler integrators* using fast and stable numerical algorithms for accelerating large-scale KCM-based protein folding simulations.

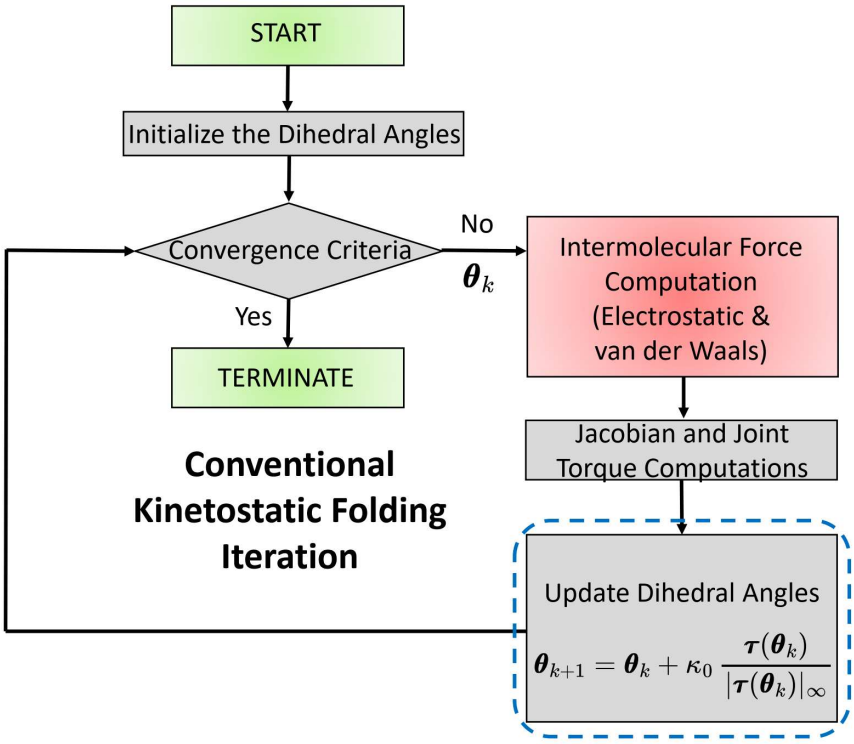

Figure 3. Flowchart of the conventional kinetostatic folding iteration. The variation of dihedral angles in the flowchart (within the dashed borders) is governed by Equation (8). The most computationally intensive procedure at each conformation of the protein molecule (highlighted in red) consists of the electrostatic and van der Waals force computations.

3. Problem Statement: Protein Folding Pathway Computation Problem (PFPCP)

In this section, we consider the kinetostatic folding of protein molecules and formulate the main problem addressed in the paper.

Protein Folding Pathway Computation Problem (PFPCP): Consider a protein backbone chain with $N-1$ peptide planes and dihedral angle vector $\boldsymbol{\theta}$, as provided by Equation (1). Furthermore, consider the KCM-based dynamics of protein folding provided by the following initial value problem:

$$\dot{\boldsymbol{\theta}} = \mathcal{F}^{\mathrm{cl}}(\boldsymbol{\theta}),\ \boldsymbol{\theta}(0) = \boldsymbol{\theta}_0 \tag{9}$$

where $\mathcal{F}^{\mathrm{cl}}(\boldsymbol{\theta}) := \frac{\boldsymbol{\tau}(\boldsymbol{\theta})}{|\boldsymbol{\tau}(\boldsymbol{\theta})|_\infty}$, in which the kinetostatic folding torque vector $\boldsymbol{\tau}(\boldsymbol{\theta})$ is provided by Equation (7). Moreover, the dihedral angle vector $\boldsymbol{\theta}_0$ is an unfolded initial conformation that belongs to the domain of attraction of a folded conformation $\boldsymbol{\theta}_f$. We devise a numerical integration scheme to integrate Equation (9) to the steady state, namely, the folded conformation $\boldsymbol{\theta}_f$, and obtain the sequence of protein folding pathway samples $\{\boldsymbol{\theta}_k\}_{k=0}^{N_s}$, where N_s is the smallest integer satisfying $|\mathcal{F}^{\mathrm{cl}}(\boldsymbol{\theta}_k)|_2 < \tau_{\mathrm{tol}}$, for all $k \geq N_s$.

Remark 1. *As discussed in Section 2.2, any folded protein conformation $\boldsymbol{\theta}_f$ is an asymptotically stable equilibrium point of the KCM nonlinear dynamical system in Equation (9) (see, e.g., [5–7]). This signifies the protein molecule's kinetostatic stationarity in folded conformations, where the absence of net kinetostatic torque reflects the balanced internal forces within the protein structure. Therefore, at any folded conformation $\boldsymbol{\theta}_f$, we have*

$$\mathcal{F}^{cl}(\boldsymbol{\theta}_f) = \mathbf{0}, \tag{10}$$

which provides a convergence criterion for KCM iterations. Specifically, given a desired tolerance τ_{tol}, if N_s is the smallest integer that satisfies

$$|\mathcal{F}^{cl}(\boldsymbol{\theta}_k)|_2 < \tau_{tol}, \text{ for all } k \geq N_s, \tag{11}$$

then N_s represents the terminal iteration step number in Equation (8) at which the aggregated free energy of the molecule converges to a sufficiently close vicinity of a free-energy-landscape local minimum. The tolerance should be chosen according to the accuracy required for the protein's folded structure and the computational efficiency; too high a value could result in premature convergence before the protein reaches its stable conformation, while too low a value could unnecessarily prolong the computation time.

In this paper we use the norm of the vector $|\mathcal{F}^{\mathrm{cl}}(\boldsymbol{\theta}_k)|_2$ to assess convergence. As the theoretical justification for such a convergence criterion, we remark that many gradient-based optimization methods (see, e.g., [55]) have relied on the norm of the gradient as a widely used criterion to determine convergence. This criterion is based on the principle that at a local minimum, the gradient of the objective function should approach zero. Similarly, in our method we use the norm $|\mathcal{F}^{\mathrm{cl}}(\boldsymbol{\theta}_k)|_2$ to assess convergence. When this norm becomes sufficiently small (below a predefined threshold τ_{tol}), it indicates that the system has reached a point where further changes in the dihedral angles are negligible, signaling that the protein has reached its folded state. The value of τ_{tol} is selected to balance computational efficiency and accuracy while avoiding premature termination of the iterations.

The *conventional approach* [5,6,12,51] for solving PFPCP relies on using the explicit Euler integration scheme with a fixed step size κ_0, resulting in the successive kinetostatic folding iteration in Equation (8). As discussed in Section 2.3, selection of the step size κ_0 in Equation (8) dictates a tradeoff between the accuracy and stability of the predicted folding pathway samples and the number of integration steps.

To prevent numerical instability and large dihedral angle variations, in the explicit Euler scheme it is necessary to choose a sufficiently small integration time step κ_0. This inevitably results in an excessive number of iterations, requiring kinetostatic folding torque computations with a large computational burden; in other words, the larger the number of iterations N_s for convergence to a folded conformation of a protein molecule with N_a atoms, the larger the needed number of interatomic electrostatic and van der Waals force calculations, which is of order $O(N_s \cdot N_a^2)$ (see Section 2.3 for further details). Moreover, to encode more physical constraints in protein folding numerical simulations, it is necessary to perform more calculations in each iteration, which grows with the number of steps N_s for convergence to the folded conformations. For instance, to encode entropy loss constraints [12], it is necessary to solve a box-constrained convex QP at each conformation of the protein molecule. The computational cost of such a convex QP using a state-of-the-art interior-point QP solver is of order $O(N^3)$ [54]. Therefore, the computational cost for encoding the entropy loss constraints through the whole kinetostatic folding simulations will be of order $O(N_s \cdot N^3)$.

To address this issue in PFPCP, we present our ΨTC-based numerical integration scheme in the next section.

Remark 2. *In all-atom molecular dynamics simulations of protein folding, symplectic integrators [56,57] offer numerical stability and geometric advantages over methods that rely on a fixed step size. However, their application is impeded in KCM-based folding simulations due to the KCM framework's reliance on the negligible role of inertial forces compared to electrostatic and van der Waals interactions [18,19,51,53]. The absence of inertial effects renders symplectic integrators inapplicable in this context.*

4. Explicit ΨTC Numerical Integration for KCM-Based Protein Folding

Our goal in this section is to solve the PFPCP problem outlined in Section 3. We introduce an explicit ΨTC numerical integration scheme with an adaptive step size tailored to KCM-based protein folding to solve the PFPCP. We begin by presenting the explicit ΨTC solution to the PFPCP under a fixed step size and analyzing its convergence as well as stability properties in Section 4.1. Thereafter, in Section 4.2 we develop a step size adaptation rule based on the widely used switched evolution relaxation (SER) technique [41]. The flowchart of our ΨTC scheme is depicted in Figure 4 and elaborated in Sections 4.1 and 4.2.

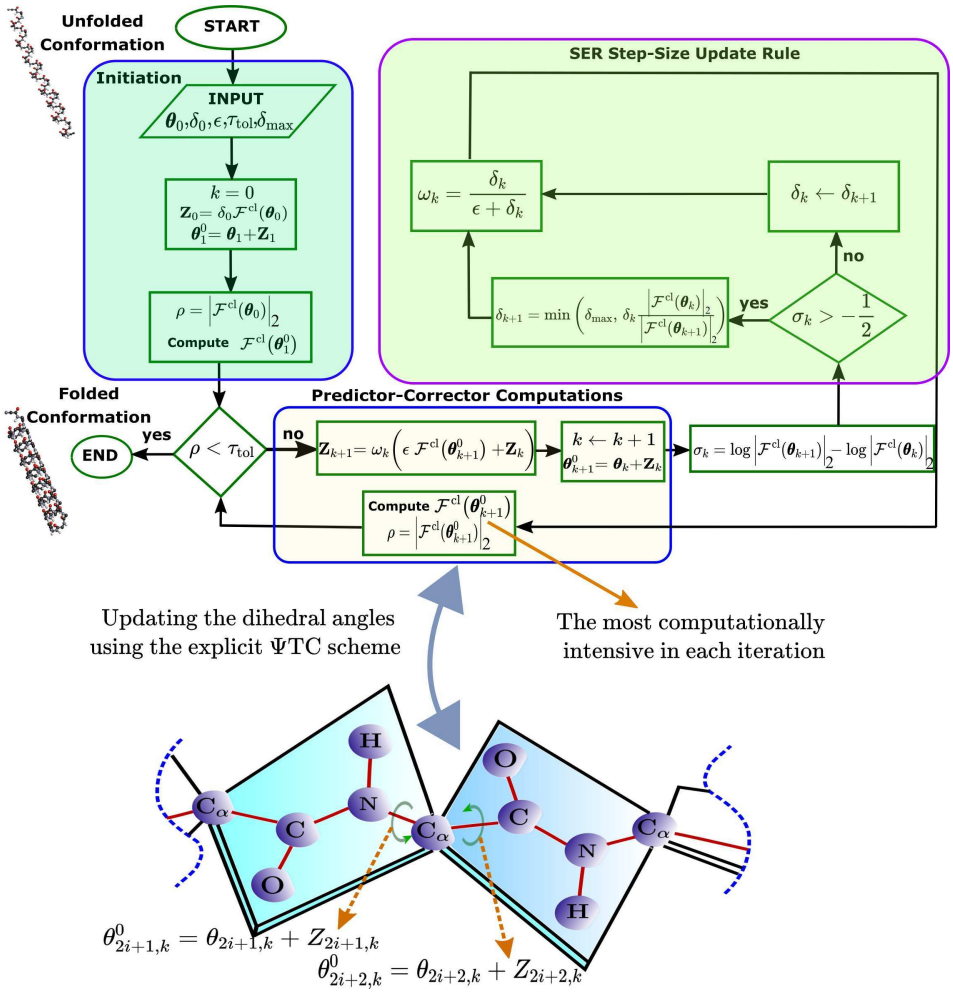

Figure 4. Flowchart of the proposed explicit ΨTC integrator for computing protein folding pathways in kinetostatic folding simulations. The ΨTC algorithm consists of four main steps: (Step 1) Initiation; (Step 2) Predictor–Corrector Computations; (Step 3) Checking Convergence; and, (Step 4) SER-based Step Size Update. In the ΨTC integration scheme with a fixed step size, (Step 4) is skipped.

4.1. Explicit ΨTC Numerical Integration with Fixed Step Size

Let us consider the initial value problem in Equation (9) associated with the PFPCP formulated in Section 3. The explicit ΨTC integration with fixed step size for solving the PFPCP can be described using the following steps (see the flowchart in Figure 4).

(Step 1) Initiation: We start with choosing a step size δ_0, a positive constant ϵ, and a given tolerance τ_{tol}. Next, we consider the iterator k and set it equal to 0. Moreover, we initiate the **internal state** of the ΨTC scheme by assigning $\delta_0 \mathcal{F}^{\text{cl}}(\boldsymbol{\theta}_0)$ to $\mathbf{Z}_0$. Note that $\mathbf{Z}_k$ is the ΨTC internal state vector in the k^{th}, $k \geq 0$, iteration which is updated along with the protein molecule configuration vector $\boldsymbol{\theta}_k$.

(Step 2) Predictor–Corrector Computation: After initiation, we run the ΨTC scheme through the following predictor–corrector numerical iteration:

$$\boldsymbol{\theta}^0_{k+1} = \boldsymbol{\theta}_k + \mathbf{Z}_k, \tag{12a}$$

$$\mathbf{Z}_{k+1} = \omega_k \Big(\epsilon \, \mathcal{F}^{\mathrm{cl}}(\boldsymbol{\theta}^0_{k+1}) + \mathbf{Z}_k \Big), \tag{12b}$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \mathbf{Z}_{k+1}, \tag{12c}$$

where

$$\omega_k := \tfrac{\delta_k}{\epsilon+\delta_k} \ \text{ for all } k \geq 0. \tag{13}$$

In the ΨTC scheme with a fixed step size, where $\delta_k = \delta_0$ and $\omega_k = \frac{\delta_0}{\epsilon+\delta_0}$, the step size δ_k is fixed for all iterator values k; see Section 4.2 for updating the step size at each iteration using an adaptive strategy. In other words, the SER update step (Step 4 in Section 4.2) is skipped in the ΨTC integration with a fixed step size. Additionally, note that there is only one folding vector field $\mathcal{F}^{\mathrm{cl}}(\cdot)$ computation at each predictor–corrector step of the ΨTC iteration in (12).

(Step 3) Checking Convergence: After each iteration, the convergence criterion is checked by comparing $\rho := \big|\mathcal{F}^{\mathrm{cl}}(\boldsymbol{\theta}^0_{k+1})\big|_2$ against the given desired tolerance τ_{tol}. The iteration is terminated when $\rho < \tau_{\mathrm{tol}}$.

Convergence Properties of the Explicit ΨTC Scheme with Fixed Step Size: Considering the kinetostatic protein folding dynamics in Equation (9) and the asymptotic stability of the folded conformation $\boldsymbol{\theta}_f$, our proposed ΨTC numerical integrator computes the protein folding pathway from a given initial unfolded conformation $\boldsymbol{\theta}_0$ in the dihedral angle space. To establish the convergence properties of the explicit ΨTC scheme in (12), we leverage the following key properties of the underlying KCM-based folding dynamics as detailed in the literature (e.g., [5,6,12,13,51]):

P1 Any folded conformation $\boldsymbol{\theta}_f$ is a locally asymptotically stable equilibrium for (9).

P2 Given any folded conformation $\boldsymbol{\theta}_f$, the folding vector field $\mathcal{F}^{\mathrm{cl}}(\cdot)$ is uniformly bounded and uniformly Lipschitz continuously differentiable in a neighborhood $\mathcal{N}_{\boldsymbol{\theta}_f}$ of $\boldsymbol{\theta}_f$.

Under Properties P1 and P2 and according to Theorem 2.1 of [38], the explicit ΨTC scheme for Equation (9) with fixed step size enjoys the following properties:

C1 For any given positive constant ϵ in the explicit ΨTC scheme with fixed step size, any $\Delta_0 > 0$, and for any sufficiently small fixed step-size $\delta_0 > 0$ in (12), there is an integer N_s such that $|\boldsymbol{\theta}_{N_s} - \boldsymbol{\theta}_f|_p < \Delta_0$, $|\boldsymbol{\theta}^0_{N_s} - \boldsymbol{\theta}_f|_p < \Delta_0$, and $|\mathbf{Z}_{N_s}|_p < \Delta_0$, where $|\cdot|_p$ is an arbitrary p-norm on $\mathbb{R}^{2N}$ (where $p \geq 1$). Furthermore, $\mathcal{F}^{\mathrm{cl}}(\boldsymbol{\theta}_k)$ is uniformly bounded for all $0 \leq k \leq N_s$.

C2 If, in addition to P1 and P2, the Jacobian of the folding torque vector field $\mathcal{F}^{\mathrm{cl}}(\cdot)$ has negative real eigenvalues at the folded conformation $\boldsymbol{\theta}_f$, and if the positive constant ϵ also satisfies $\epsilon\varrho\big(\frac{\partial \mathcal{F}^{\mathrm{cl}}}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta}_f)\big) < \frac{4}{3}$, then there exists a p^*-norm, namely, a norm $|\cdot|_{p^*}$ on $\mathbb{R}^{4N}$, such that $[\boldsymbol{\theta}_k, \mathbf{Z}_k]^\top$ converges q-linearly to $[\boldsymbol{\theta}_f, \mathbf{0}]^\top$ (see the **Notation** in Section 1).

Property C1 guarantees that for any given prescribed accuracy Δ_0 it is always possible to choose a small enough step size δ_0 such that the protein configuration converges to an arbitrarily small vicinity of the folded molecule conformation determined by Δ_0. Furthermore, the internal state of the integrator is guaranteed to remain bounded. Additionally, according to Property C2, if a certain spectral radius inequality condition is satisfied, then the protein configuration computed by the numerical integrator is guaranteed to converge to the folded conformation with a q-linear rate (see the **Notation** in Section 1).

4.2. Switched Evolution Relaxation (SER) Step Size Adaptation Rule

To dynamically change the step size of the ΨTC integrator in each iteration, we adopt the SER method originally introduced by Mulder and van Leer [41] and subsequently employed

within the ΨTC framework [35,36]. The intuition underlying the SER step size adaptation rule is that when the initial step sizes are small, the integration is close to the explicit Euler scheme with fixed step size. When near the folded protein configuration $\boldsymbol{\theta}_f$, the time step safely grows while keeping the protein molecule conformations within the domain of attraction of $\boldsymbol{\theta}_f$. The details of the SER step size update rule can be stated as follows:

(Step 4) Step Size Update: We monitor the logarithmic change in the folding vector field $\mathcal{F}^{\text{cl}}(\cdot)$ from each iteration k to the next one $k+1$ by computing the logarithmic growth monitoring variable σ_k according to

$$\sigma_k = \log \left|\mathcal{F}^{\text{cl}}(\boldsymbol{\theta}_{k+1})\right|_2 - \log \left|\mathcal{F}^{\text{cl}}(\boldsymbol{\theta}_k)\right|_2. \tag{14}$$

If the logarithmic change in the folding vector field satisfies $\sigma_k > -\frac{1}{2}$, then we update the integrator step-size using

$$\delta_{k+1} = \min\left(\delta_{\max}, \delta_k \frac{\left|\mathcal{F}^{\text{cl}}(\boldsymbol{\theta}_k)\right|_2}{\left|\mathcal{F}^{\text{cl}}(\boldsymbol{\theta}_{k+1})\right|_2}\right), \tag{15}$$

where $\delta_{\max} > 0$ is a design parameter that sets an upper limit on the growth of step sizes in the explicit ΨTC algorithm. On the other hand, if the logarithmic change in the folding vector field satisfies $\sigma_k \leq -\frac{1}{2}$, then we do not update the step size. △

Step 4 dynamically adjusts the step size δ_k at each iteration. In particular, it requires that the logarithmic change in the protein folding vector field $\mathcal{F}^{\text{cl}}(\cdot)$ be monitored. When this logarithmic change satisfies the inequality in (14), then the step size δ_k is updated. Accordingly, the parameter ω_k provided by (13) is adjusted. This parameter is then utilized in the correction step of the predictor–corrector computations in (12b).

The SER step size update strategy prioritizes capturing critical transient conformations early in the protein folding process while gradually increasing the step size as the molecule approaches the folded conformation $\boldsymbol{\theta}_f$. In the SER update rule provided by (15), comparison with $\delta_{\max}$ safeguards the step size δ_k against becoming too large. Furthermore, the logarithmic monitoring variable σ_k prevents the steps from very rapid oscillations. Additionally, as stated in the proof of Theorem 2.1, the early iterations of the ΨTC scheme in [38] differ from its explicit Euler counterpart by $O(\delta_0^2)$, where the constant in the O-term is merely dependent on the folding vector field $\mathcal{F}^{\text{cl}}(\cdot)$ in a neighborhood of the initial unfolded conformation $\boldsymbol{\theta}_0$.

Remark 3. *The proposed ΨTC method in this paper falls within the category of explicit ΨTC schemes (see, e.g., [38,39]). Explicit ΨTC schemes are generally preferred over implicit ones for protein folding pathway computations because they circumvent the need to solve linear systems. In contrast, implicit ΨTC schemes (see, e.g., [35,36]) require solving such systems, which is computationally expensive in this context due to the necessity of calculating the Jacobian of the nonlinear folding torque vector field $\mathcal{F}^{cl}(\cdot)$ at each conformation along the pathway.*

5. Numerical Simulations

In this section, we present kinetostatic protein folding simulation results that validate our proposed ΨTC numerical integration scheme for computing both transient and final protein conformations during the kinetostatic folding process. The simulations benchmark the performance of our ΨTC integrator against the explicit Euler integration scheme, which is the only integrator used thus far in the kinetostatic folding literature [5,6,12,51]. Our kinetostatic folding simulations examine how unfolded protein conformations converge towards α-helix secondary structures [58]. The α-helix structure is characterized by a tightly coiled arrangement of the amino acid chain, which is a prevalent structural motif in proteins.

We conducted our simulations following the guidelines of Protofold I [6,51], utilizing an Intel® Core™ i7-6770HQ CPU at 2.60GHz. In our simulations, we considered *three scenarios* with protein molecule backbone chains consisting of 15, 30, and 50 peptide planes.

These choices result in dihedral angle spaces with 32, 62, and 102 dimensions, respectively. These three protein backbone chains have $N_a = 78$, $N_a = 153$, and $N_a = 253$ atoms, respectively. As elaborated in Section 2.2, we let the backbone chain fold kinetostatically under the influence of the interatomic force vector field associated with the initial value problem provided in Equation (9).

Using the three aforementioned protein backbone chains, we carried out *two distinct groups* of kinetostatic protein folding simulations associated with ΨTC and explicit Euler integrator schemes. In the first set, we simulated the folding pathways of the three protein backbone chains using the explicit ΨTC integrator described in Section 4. The integration scheme for ΨTC is illustrated in the flowchart in Figure 4. The ΨTC integrator design parameters were ϵ, δ_0, and $\delta_{\max}$, which are provided in Table 2.

Table 2. Parameters of the integration schemes associated with the kinetostatic protein folding simulations.

dim $\mathcal{Q}$	ΨTC (δ_0)	ΨTC ($\frac{\delta_{\max}}{\delta_0}$)	ΨTC ($\frac{\delta_0}{\epsilon}$)	Euler-1 ($\frac{\kappa_0}{\delta_0}$)	Euler-2 ($\frac{\kappa_0}{\delta_0}$)
32	2×10^{-4}	5	2×10^{-4}	1	0.5
62	1×10^{-4}	5	1×10^{-4}	1	0.5
102	1×10^{-4}	5	1×10^{-4}	1	0.5

Figures 5–7 present the kinetostatic folding simulation results from the explicit ΨTC integration schemes applied to each of the three protein backbone chains. These figures show the free energy $\mathcal{G}(\boldsymbol{\theta})$ in kcal/mol for the backbone chain of protein molecules with dihedral angle vectors of dimensions 32, 62, and 102, respectively, along with their transient conformations along the folding pathway. Additionally, the figures illustrate the step size evolution δ_k for the explicit ΨTC scheme in each of the three cases. The step size δ_k was updated according to the SER update rule provided by Equation (15). As the protein molecules approach their final folded conformations, the step size δ_k increases and eventually saturates at $\delta_{\max}$, which is the safeguard design parameter in the SER update rule.

Next, we performed kinetostatic folding simulations on the aforementioned three backbone chains using the explicit Euler scheme provided by Equation (8) with two different sets of fixed step sizes κ_0. The initial protein conformations were all chosen to be the same pre-coiled backbone chains as their counterparts in the ΨTC-based numerical simulations; in other words, to guarantee uniform starting conditions across both groups of numerical integration schemes, identical initial conformations for the protein backbone chains were adopted as pre-coiled structures near the α-helix configurations.

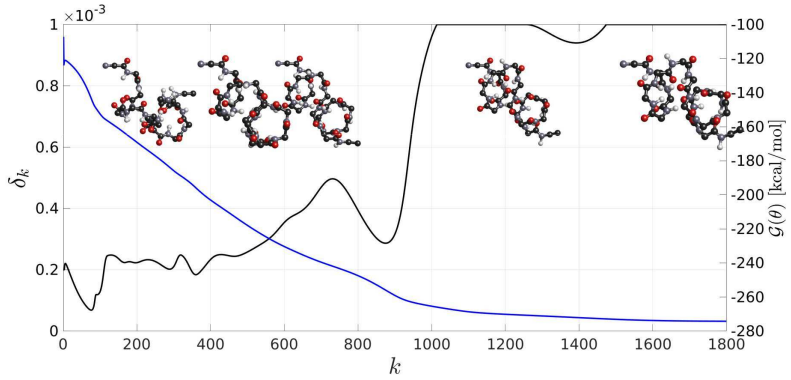

Figure 5. Simulation results associated with the ΨTC integrator. The free energy of the backbone chain of a protein molecule with a 32-dimensional dihedral angle vector (blue curve; $\mathcal{G}(\boldsymbol{\theta})$ on the right axis), its transient conformations along the folding pathway, and the step size of the explicit ΨTC scheme (black curve; δ_k on the left axis). From left to right, the five plotted protein backbone chain conformations correspond to iteration numbers $k = 100$, $k = 300$, $k = 500$, $k = 1000$, and $k = 1400$, respectively.

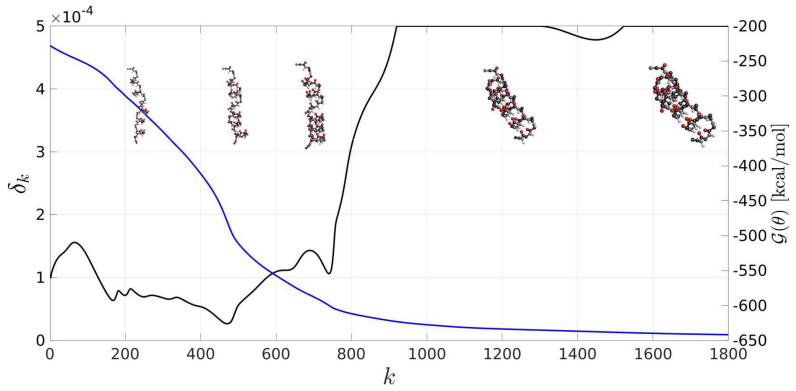

Figure 6. Simulation results associated with the ΨTC integrator. The free energy of the backbone chain of a protein molecule with a 62-dimensional dihedral angle vector (blue curve; $\mathcal{G}(\boldsymbol{\theta})$ on the right axis), its transient conformations along the folding pathway, and the step size of the explicit ΨTC scheme (black curve; δ_k on the left axis). From left to right, the five plotted protein backbone chain conformations correspond to iteration numbers $k = 100$, $k = 300$, $k = 500$, $k = 1000$, and $k = 1400$, respectively.

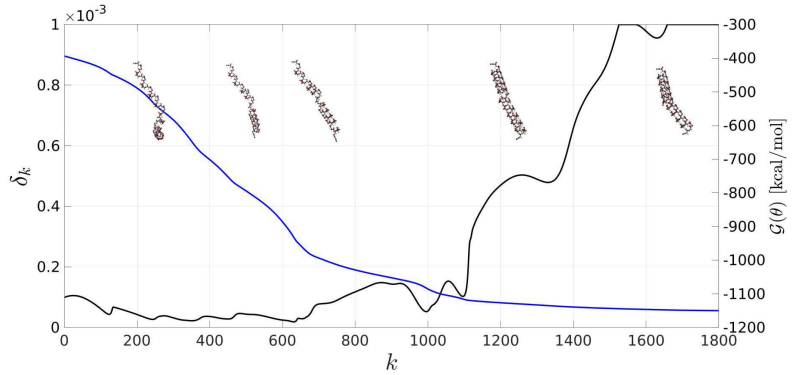

Figure 7. Simulation results associated with the ΨTC integrator. The free energy of the backbone chain of a protein molecule with a 102-dimensional dihedral angle vector (blue curve; $\mathcal{G}(\boldsymbol{\theta})$ on the right axis), its transient conformations along the folding pathway, and the step size of the explicit ΨTC scheme (black curve; δ_k on the left axis). From left to right, the five plotted protein backbone chain conformations correspond to iteration numbers $k = 100$, $k = 300$, $k = 500$, $k = 1000$, and $k = 1400$, respectively.

We compared the obtained results from our proposed ΨTC scheme against the explicit Euler integrator method with a fixed step size. Two different step sizes were chosen for the explicit Euler schemes; the first one was set equal to the initial step size for the explicit ΨTC scheme, namely, $\frac{\kappa_0}{\delta_0} = 1$ (see the Euler-1 column in Table 2), while the second step-size was set equal to half of the initial step-size for the explicit ΨTC scheme, namely, $\frac{\kappa_0}{\delta_0} = 0.5$ (see the Euler-2 column in Table 2). As discussed in Section 2.3, the size of κ_0 in the explicit Euler integrator dictates a tradeoff between the stability and accuracy of the obtained folding pathways and the convergence speed of the explicit Euler schemes.

The plots in Figure 8 illustrate the free energy $\mathcal{G}(\boldsymbol{\theta})$ in kcal/mol of the backbone chain of these protein molecules with dihedral angle vectors of dimension 32, 62, and 102, respectively. As can be seen from the figures, in the case of $\frac{\kappa_0}{\delta_0} = 1$ (i.e., explicit Euler with a larger step size), the oscillations and instability in the protein free energy profile computed from the explicit Euler scheme (the continuous curves) are due to the large step size chosen for this method. In the case of $\frac{\kappa_0}{\delta_0} = 0.5$ (i.e., the explicit Euler with a smaller step size), the protein free energy (the dash-dot-dashed curves) does not suffer such oscillations, but struggles to converge to the local minimum associated with the α-helix folded structure even after 1800 iterations. These results are in contrast to the stable and faster convergence of the ΨTC-based results, where convergence takes place around iteration $k = 1400$.

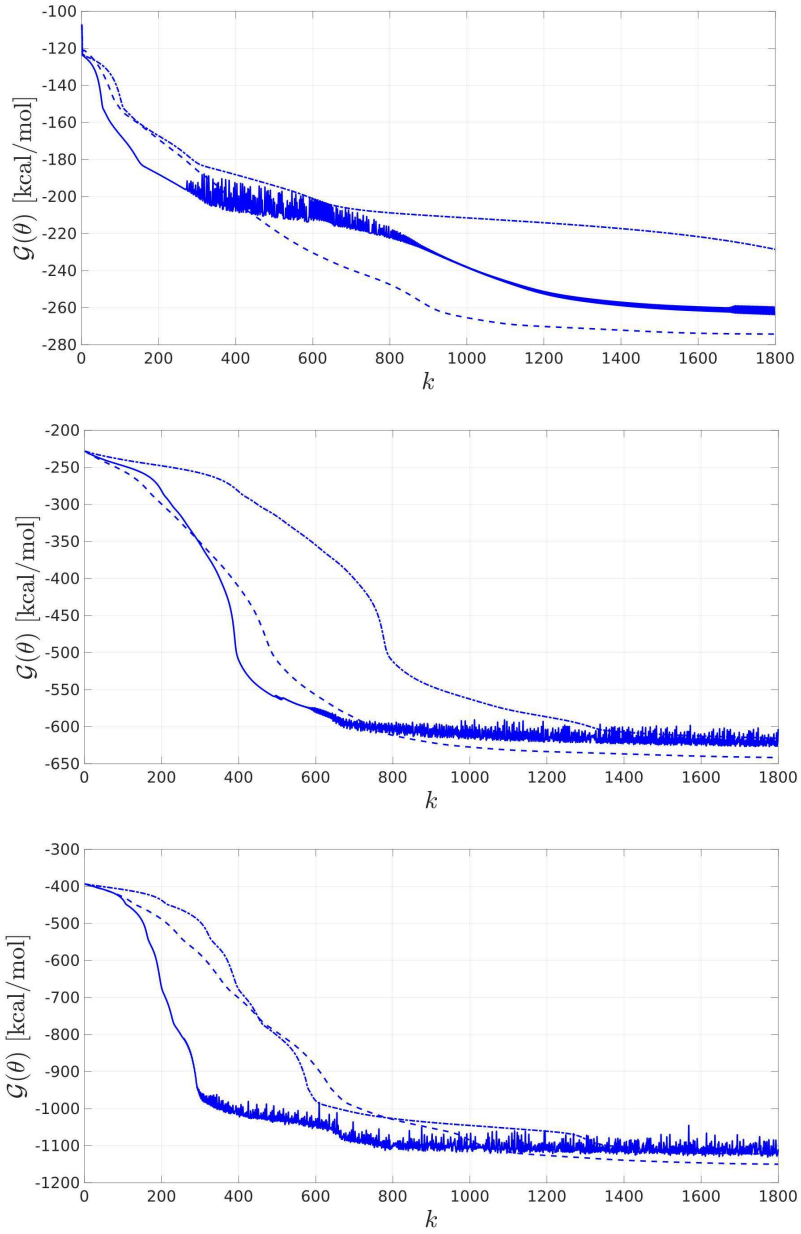

Figure 8. Comparison of the results obtained from our proposed ΨTC scheme against the explicit Euler integrator method with a fixed step size. The free energy of the backbone chain of protein molecules obtained from the explicit ΨTC scheme (dashed curve), explicit Euler scheme with step size satisfying $\frac{\kappa_0}{\delta_0} = 1$ (continuous curve), and explicit Euler scheme with step size satisfying $\frac{\kappa_0}{\delta_0} = 0.5$ (dash-dot-dashed curve). In the case of $\frac{\kappa_0}{\delta_0} = 1$, the oscillations and instability in the protein free energy profile computed from the explicit Euler scheme (the continuous curves) are due to the large step size chosen for this method. In the case of $\frac{\kappa_0}{\delta_0} = 0.5$ (the dash-dot-dashed curves), the protein free energy does not suffer such oscillations, but struggles to converge to its local minimum even after 1800 iterations.

Finally, another notable fact in the simulations is the proximity of the obtained results during the first few steps of these three different numerical schemes. Indeed, using the proof in [38], Theorem 2.1, we can state that the early iterations of the ΨTC scheme differ from its explicit Euler counterparts by $O(\delta_0^2)$, where the constant in the O-term is merely dependent on the folding vector field $\mathcal{F}^{\mathrm{cl}}(\cdot)$ in a neighborhood of the initial unfolded conformation $\boldsymbol{\theta}_0$. The relationship between the accuracy of the iterative computation and the speed of convergence in our ΨTC method is closely tied to the adaptive step size

mechanism. Higher accuracy requires smaller values of the stopping threshold τ_{tol}, which increases the number of iterations. However, our adaptive step size approach dynamically adjusts the step size to accelerate convergence in the early stages and ensure accuracy in later stages, balancing the tradeoff between accuracy and computational efficiency.

6. Conclusions

In this paper, we have considered protein molecules modeled as nanomechanisms with hyper degrees of freedom consisting of numerous rigid peptide plane linkages. To compute the protein transient and final folded conformations during kinetostatic folding, we propose an explicit ΨTC numerical integration technique with step size adaptation tailored to the underlying kinetostatic folding. Our proposed numerical scheme departs from the established literature on the KCM framework, where numerical simulations of kinetostatic protein folding have exclusively relied on explicit Euler schemes with a fixed step size. Moreover, this paper provides the numerical stability and convergence properties of the developed explicit ΨTC numerical algorithm in the context of kinetostatic protein folding. Thanks to its convergence and stability properties, our proposed ΨTC numerical algorithm can reduce the computational burden on kinetostatic folding numerical simulations.

Our current simulations focused primarily on in vacuo scenarios and excluded protein side chains, which simplifies the model. Future work will look to include these aspects in order to better capture the full dynamics of protein folding. Additionally, while our method improves computational efficiency, its performance in real-life applications for very large protein systems could be further optimized. We intend to explore this in our future studies, particularly examining the performance of the integrator in solute to mimic more realistic biological conditions. As future research directions, we envision that the proposed methodology will have potential applications for efficient numerical investigation of kinetostatic protein folding under solvation effects and entropy loss constraints as well as in the design and numerical simulation of protein-based nanorobots/nanomachines. Another direction for future research includes investigating the β-pleated sheet structure, which plays a crucial role in protein folding by facilitating the reorientation of the polypeptide chain. Remarkably, even small proteins such as chignolin, a synthetic mini-protein consisting of just ten amino acids, can demonstrate robust beta-hairpin folding.

In addition to kinetostatic protein folding, our proposed ΨTC-based scheme has *potential applications* for developing high-fidelity numerical integrators for soft robotic mechanisms [59–61]. A frequently encountered issue in soft robot arms and cable-driven soft locomotive mechanisms involves numerical integration of the implicit nonlinear differential equations resulting from their elastica models (see, e.g., [60,61]). Remarkably, the protein kinematic structure in the KCM framework is exactly the same as robotic manipulators with hyper degrees of freedom, as described in the work of Mochiyama et al. (see, e.g., [45]). This type of kinematic modeling has also been used for multisection continuum robots (see, e.g., [46]). Therefore, the methodology developed in this paper is also applicable to such multisection continuum robotic mechanisms with nonlinear and high-dimensional joint space dynamics, provided that their closed-loop dynamics satisfy the same continuity and boundedness properties as the KCM-based protein folding vector field discussed here. The potential of this integrator in robotics and control systems is particularly promising, but relatively unexplored thus far. To date, apart from the work presented in this paper, one notable application in the robotics domain has been detailed in [62]. Their study conducted numerical simulations on the model predictive control (MPC) of a quadrotor, The results highlight the ability of our proposed ΨTC integrator, which can significantly reduce computational times compared to the traditional quadratic programming solvers utilized in MPC schemes.

Author Contributions: Conceptualization, A.M.; methodology, A.M.; software, A.M.; formal analysis, A.K., K.Z. and A.M.; investigation, A.M.; resources, A.M.; data curation, A.K. and K. Z.; writing—original draft preparation, A.K., K.Z. and A. M.; writing—review and editing, A.M.; visualization, A.K. and K. Z.; supervision, A.M.; project administration, A.M.; funding acquisition, A.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Science Foundation (NSF) through award number CMMI-2153744.

Data Availability Statement: Dataset available on request from the authors

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ling, M.; Howell, L.L.; Cao, J.; Chen, G. Kinetostatic and dynamic modeling of flexure-based compliant mechanisms: A survey. *Appl. Mech. Rev.* **2020**, *72*, 030802. [CrossRef]
2. Firouzeh, A.; Paik, J. An under-actuated origami gripper with adjustable stiffness joints for multiple grasp modes. *Smart Mater. Struct.* **2017**, *26*, 055035. [CrossRef]
3. Lilge, S.; Burgner-Kahrs, J. Kinetostatic modeling of tendon-driven parallel continuum robots. *IEEE Trans. Robot.* **2022**, *39*, 1563–1579. [CrossRef]
4. Childs, J.A.; Rucker, C. A Kinetostatic Model for Concentric Push–Pull Robots. *IEEE Trans. Robot.* **2024**, *40*, 554–572. [CrossRef] [PubMed]
5. Tavousi, P.; Behandish, M.; Ilieş, H.T.; Kazerounian, K. Protofold II: Enhanced model and implementation for kinetostatic protein folding. *ASME J. Nanotechnol. Eng. Med.* **2015**, *6*, 034601. [CrossRef]
6. Tavousi, P. On the Systematic Design and Analysis of Artificial Molecular Machines. Ph.D. Thesis, University of Connecticut, Mansfield, CT, USA, 2016.
7. Mohammadi, A.; Al Janaideh, M. Sign gradient descent algorithms for kinetostatic protein folding. In Proceedings of the 2023 International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS), Abu Dhabi, United Arab Emirates, 9–13 October 2023; pp. 1–6. [CrossRef]
8. Kazerounian, K.; Latif, K.; Rodriguez, K.; Alvarado, C. Nano-kinematics for analysis of protein molecules. *ASME J. Mech. Des.* **2005**, *127*, 699–711. [CrossRef]
9. Gohil, M.K.; Chakraborty, A.; Dasgupta, B. Hyper-redundant robots and bioinformatics: Modelling loops in RNA. In Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, Hungary, 9–12 October 2016; pp. 3694–3700. [CrossRef]
10. Diez, M.; Petuya, V.; Martínez-Cruz, L.A.; Hernández, A. Biokinematic protein simulation by an adaptive dihedral angle approach. *Mech. Mach. Theory* **2013**, *69*, 105–114. [CrossRef]
11. Ekenna, C.; Thomas, S.; Amato, N.M. Adaptive local learning in sampling-based motion planning for protein folding. *BMC Syst. Biol.* **2016**, *10*, 165–179. [CrossRef]
12. Mohammadi, A.; Spong, M.W. Quadratic optimization-based nonlinear control for protein conformation prediction. *IEEE Control Syst. Lett.* **2022**, *6*, 373–378. [CrossRef]
13. Mohammadi, A.; Spong, M.W. Chetaev Instability Framework for Kinetostatic Compliance-Based Protein Unfolding. *IEEE Control Syst. Lett.* **2022**, *6*, 2755–2760. [CrossRef]
14. Maruyama, Y.; Mitsutake, A. Analysis of structural stability of chignolin. *J. Phys. Chem. B* **2018**, *122*, 3801–3814. [CrossRef] [PubMed]
15. Madden, C.; Bohnenkamp, P.; Kazerounian, K.; Ilieş, H.T. Residue level three-dimensional workspace maps for conformational trajectory planning of proteins. *Int. J. Robot. Res.* **2009**, *28*, 450–463. [CrossRef]
16. Lee, S.; Chirikjian, G.S. Pose Analysis of Alpha-Carbons in Proteins. *Int. J. Robot. Res.* **2005**, *24*, 183–210. [CrossRef]
17. Arkun, Y.; Gür, M. Protein folding using coarse-grained optimal control and molecular dynamics. *IFAC Proc. Vol.* **2011**, *44*, 14213–14216. [CrossRef]
18. Arkun, Y.; Erman, B. Prediction of optimal folding routes of proteins that satisfy the principle of lowest entropy loss: Dynamic contact maps and optimal control. *PLoS ONE* **2010**, *5*, e13275. [CrossRef]
19. Arkun, Y.; Gür, M. Combining optimal control theory and molecular dynamics for protein folding. *PLoS ONE* **2012**, *7*, e29628. [CrossRef]
20. Kazerounian, K.; Ilies, H. The Evolving Role of Robot Kinematics in Bio-Nanotechnology. In Proceedings of the International Symposium on Advances in Robot Kinematics, Bilbao, Spain, 30 June–4 July 2024; pp. 77–87. [CrossRef]
21. Shahbazi, Z.; Ilieş, H.T.; Kazerounian, K. Hydrogen bonds and kinematic mobility of protein molecules. *ASME J. Mech. Robot.* **2010**, *2*, 021009. [CrossRef]
22. Chorsi, M.T.; Tavousi, P.; Mundrane, C.; Gorbatyuk, V.; Ilieş, H.; Kazerounian, K. One Degree of Freedom 7-R Closed Loop Linkage as a Building Block of Nanorobots. In *Advances in Robot Kinematics 2020*; Springer: Berling/Heidelberg, Germany, 2020; pp. 41–48.

23. Chorsi, M.T.; Tavousi, P.; Mundrane, C.; Gorbatyuk, V.; Kazerounian, K.; Ilies, H. Kinematic design of functional nanoscale mechanisms from molecular primitives. *J. Micro-Nano-Manuf.* **2021**, *9*, 021005. [CrossRef]
24. Mundrane, C.; Chorsi, M.; Vinogradova, O.; Ilieş, H.; Kazerounian, K. Exploring electric field perturbations as the actuator for nanorobots and nanomachines. In Proceedings of the International Symposium on Advances in Robot Kinematics, Bilbao, Spain, 26–30 June 2022; pp. 257–265. [CrossRef]
25. Chorsi, M.S.; Linthicum, W.; Pozhidaeva, A.; Mundrane, C.; Mulligan, V.K.; Chen, Y.; Tavousi, P.; Gorbatyuk, V.; Vinogradova, O.; Hoch, J.C.; et al. Ultra-confined controllable cyclic peptides as supramolecular biomaterials. *Nano Today* **2024**, *56*, 102247. [CrossRef]
26. Hamdi, M.; Ferreira, A. Multiscale design and modeling of protein-based nanomechanisms for nanorobotics. *Int. J. Robot. Res.* **2009**, *28*, 436–449. [CrossRef]
27. Testard, N.J.; Chevallereau, C.; Wenger, P. Comparison of explicit and implicit numerical integrations for a tendon-driven robot. In Proceedings of the International Conference on Cable-Driven Parallel Robots 2023, Nantes, France, 25–28 June 2023; pp. 234–245. [CrossRef]
28. Gibson, C.; Murphey, T.D. Geometric integration of impact during an orbital docking procedure. In Proceedings of the 2010 IEEE International Conference on Automation Science and Engineering, Toronto, ON, Canada, 21–24 August 2010; pp. 928–932. [CrossRef]
29. Pekarek, D.; Marsden, J.E. Variational collision integrators and optimal control. In Proceedings of the 18th International Symposium on Mathematical Theory of Networks & Systems (MTNS), Blacksburg, VI, USA, 28 July–1 August 2008.
30. Fan, T.; Schultz, J.; Murphey, T. Efficient computation of higher-order variational integrators in robotic simulation and trajectory optimization. In *Algorithmic Foundations of Robotics XIII*; Springer: Berling/Heidelberg, Germany, 2018; pp. 689–706. [CrossRef]
31. Braun, D.J.; Goldfarb, M. Simulation of constrained mechanical systems—Part II: Explicit numerical integration. *J. Appl. Mech.* **2012**, *79*, 041018. [CrossRef]
32. Fang, L.; Kissel, A.; Zhang, R.; Negrut, D. On the use of half-implicit numerical integration in multibody dynamics. *ASME J. Comput. Nonlinear Dyn.* **2023**, *18*, 014501. [CrossRef]
33. Till, J.; Aloi, V.; Rucker, C. Real-time dynamics of soft and continuum robots based on Cosserat rod models. *Int. J. Robot. Res.* **2019**, *38*, 723–746. [CrossRef]
34. Nordkvist, N.; Sanyal, A.K. A Lie group variational integrator for rigid body motion in SE(3) with applications to underwater vehicle dynamics. In Proceedings of the 49th IEEE Conference on Decision and Control (CDC), Atlanta, GA, USA, 15–17 December 2010; pp. 5414–5419. [CrossRef]
35. Kelley, C.T.; Keyes, D.E. Convergence analysis of pseudo-transient continuation. *SIAM J. Numer. Anal.* **1998**, *35*, 508–523. [CrossRef]
36. Coffey, T.S.; Kelley, C.T.; Keyes, D.E. Pseudotransient continuation and differential-algebraic equations. *SIAM J. Sci. Comput.* **2003**, *25*, 553–569. [CrossRef]
37. Han, T.M.; Han, Y. Solving implicit equations arising from Adams-Moulton methods. *BIT Numer. Math.* **2002**, *42*, 336–350. :1021951025649 [CrossRef]
38. Kelley, C.; Liao, L.Z. Explicit pseudo-transient continuation. *Pac. J. Optim.* **2013**, *9*, 77–91.
39. Han, T.; Han, Y. Numerical solution for super large scale systems. *IEEE Access* **2013**, *1*, 537–544. [CrossRef]
40. Ceze, M.; Fidkowski, K.J. Constrained pseudo-transient continuation. *Int. J. Numer. Methods Eng.* **2015**, *102*, 1683–1703. [CrossRef]
41. Mulder, W.A.; van Leer, B. Experiments with implicit upwind methods for the Euler equations. *J. Comput. Phys.* **1985**, *59*, 232–246. [CrossRef]
42. Ceze, M.; Fidkowski, K. Pseudo-transient continuation, solution update methods, and CFL strategies for DG discretizations of the RANS-SA equations. In Proceedings of the 21st AIAA Computational Fluid Dynamics Conference, San Diego, CA, USA, 24–27 June 2013; p. 2686. [CrossRef]
43. Shestakov, A.I.; Offner, S.S. A multigroup diffusion solver using pseudo transient continuation for a radiation-hydrodynamic code with patch-based AMR. *J. Comput. Phys.* **2008**, *227*, 2154–2186. [CrossRef]
44. Rashidi, S.; Esfahani, J.A.; Maskaniyan, M. Applications of magnetohydrodynamics in biological systems—A review on the numerical studies. *J. Magn. Magn. Mater.* **2017**, *439*, 358–372. [CrossRef]
45. Mochiyama, H.; Shimemura, E.; Kobayashi, H. Shape control of manipulators with hyper degrees of freedom. *Int. J. Robot. Res.* **1999**, *18*, 584–600. [CrossRef]
46. Jones, B.A.; Walker, I.D. Kinematics for multisection continuum robots. *IEEE Trans. Robot.* **2006**, *22*, 43–55. [CrossRef]
47. Seleem, I.A.; El-Hussieny, H.; Ishii, H. Imitation-Based Motion Planning and Control of a Multi-Section Continuum Robot Interacting with the Environment. *IEEE Robot. Autom. Lett.* **2023**, *8*, 1351–1358. [CrossRef]
48. Bruder, D.; Fu, X.; Gillespie, R.B.; Remy, C.D.; Vasudevan, R. Data-driven control of soft robots using Koopman operator theory. *IEEE Trans. Robot.* **2020**, *37*, 948–961. [CrossRef]
49. Kacem, A.; Zbiss, K.; Mohammadi, A. A Numerical Integrator for Forward Dynamics Simulations of Folding Process for Protein Molecules Modeled as Hyper-Redundant Robots. In Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 1–5 October 2023.
50. Finkelstein, A.V.; Ptitsyn, O. *Protein Physics: A Course of Lectures*; Elsevier: Amsterdam, The Netherlands, 2016.

51. Kazerounian, K.; Latif, K.; Alvarado, C. Protofold: A successive kinetostatic compliance method for protein conformation prediction. *ASME J. Mech. Des.* **2005**, *127*, 712–717. [CrossRef]
52. Alvarado, C.; Kazerounian, K. On the rotational operators in protein structure simulations. *Prot. Eng.* **2003**, *16*, 717–720. [CrossRef]
53. Adolf, D.B.; Ediger, M.D. Brownian dynamics simulations of local motions in polyisoprene. *Macromolecules* **1991**, *24*, 5834–5842. [CrossRef]
54. Wang, Y.; Boyd, S. Fast evaluation of quadratic control-Lyapunov policy. *IEEE Trans. Control Syst. Technol.* **2010**, *19*, 939–946. [CrossRef]
55. Fliege, J.; Vaz, A.I.F.; Vicente, L.N. Complexity of gradient descent for multiobjective optimization. *Optim. Methods Softw.* **2019**, *34*, 949–959. [CrossRef]
56. Tranchida, J.; Plimpton, S.J.; Thibaudeau, P.; Thompson, A.P. Massively parallel symplectic algorithm for coupled magnetic spin dynamics and molecular dynamics. *J. Chem. Phys.* **2018**, *372*, 406–425. [CrossRef]
57. Gray, S.K.; Noid, D.W.; Sumpter, B.G. Symplectic integrators for large scale molecular dynamics simulations: A comparison of several explicit methods. *J. Chem. Phys.* **1994**, *101*, 4062–4072. [CrossRef]
58. Scholtz, J.M.; Baldwin, R.L. The mechanism of alpha-helix formation by peptides. *Annu. Rev. Biophys. Biomol. Struct.* **1992**, *21*, 95–118. [CrossRef] [PubMed]
59. Coevoet, E.; Morales-Bieze, T.; Largilliere, F.; Zhang, Z.; Thieffry, M.; Sanz-Lopez, M.; Carrez, B.; Marchal, D.; Goury, O.; Dequidt, J.; et al. Software toolkit for modeling, simulation, and control of soft robots. *Adv. Robot.* **2017**, *31*, 1208–1224. [CrossRef]
60. Armanini, C.; Dal Corso, F.; Misseroni, D.; Bigoni, D. From the elastica compass to the elastica catapult: An essay on the mechanics of soft robot arm. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **2017**, *473*, 20160870. [CrossRef]
61. Bern, J.M.; Banzet, P.; Poranne, R.; Coros, S. Trajectory optimization for cable-driven soft robot locomotion. In Proceedings of the Robotics: Science and Systems, Delft, The Netherlands, 15–19 July 2019; Volume 1. [CrossRef]
62. Calogero, L.; Pagone, M.; Rizzo, A. Enhanced Quadratic Programming via Pseudo-Transient Continuation: An Application to Model Predictive Control. *IEEE Control. Syst. Lett.* **2024**, *8*, 1661–1666. [CrossRef]