Dynamics of Closed-Chain Multibody Systems

October 7, 2026

Contents

General Introduction **iv**

General State of the Art **vi**

1 Dynamics Formulations **1**
1.1 State of the Art . . . 1
1.2 The same mechanics, different formulations . . . 1
1.2.1 Newton–Euler equations . . . 1
1.2.2 Lagrangian equations . . . 1
1.2.3 Virtual work and d'Alembert's principle . . . 2
1.2.4 Kane's method . . . 2
1.2.5 Hamiltonian formulation . . . 2
1.3 How to choose a formulation . . . 2
1.4 One double pendulum, five formulations . . . 3
1.4.1 Newton–Euler: retain the rod reactions . . . 4
1.4.2 Lagrange: derive directly in angles . . . 5
1.4.3 Virtual work and d'Alembert: project onto admissible motion . . . 6
1.4.4 Kane: use generalized speeds and partial velocities . . . 7
1.4.5 Hamiltonian: evolve positions and momenta . . . 7
1.5 Formulation used in this book . . . 8
1.6 Summary . . . 8

2 Mechanism and Modeling Assumptions **9**
2.1 State of the Art . . . 9
2.2 Open chains and closed chains . . . 9
2.3 From equation choice to physical model . . . 9
2.4 Five links, five pins, one loop . . . 9
2.5 Bodies, joints, and loads . . . 10
2.6 Modeling contract and scope . . . 11
2.7 Mobility preview . . . 11
2.8 Summary and transition . . . 11

3 Coordinates and Rigid-Body Kinematics **12**
3.1 State of the Art . . . 12
3.2 Cartesian coordinates for planar bodies . . . 12
3.3 Reference frames and planar rotation . . . 12
3.4 World position and velocity of an attachment point . . . 13
3.5 Double pendulum: forward and inverse kinematics . . . 13
3.6 Five-link loop: Cartesian and reduced kinematics . . . 14

3.7 Summary 16

4 Deriving Dynamics and Preparing an ODE 17
4.1 State of the Art 17
4.2 Coordinates and modeling assumptions 17
4.3 Deriving the equations with Lagrange's method 17
4.3.1 Positions, velocities, and energy 18
4.3.2 Euler–Lagrange residual and equation extraction 19
4.4 Preparing the first-order ODE 20
4.4.1 Language-neutral pseudocode 20
4.5 Numerical example and energy check 21
4.6 From the open-chain example to the five-link closed chain 21
4.6.1 Coordinates, pin locations, and position closure 21
4.6.2 Differentiate closure to obtain velocity and acceleration conditions . 23
4.6.3 Energies and ideal multiplier equations 24
4.6.4 From the KKT solve to an ODE, and why closure drifts 25
4.7 Summary and transition 26

5 Joint Constraints and Loop Closure 27
5.1 State of the Art 27
5.2 Revolute-pin constraint 27
5.3 Grounding and assembling the loop 27
5.3.1 Assembly procedure used by the symbolic model 28
5.4 Constraint Jacobian, rank, and local mobility 28
5.5 Summary and transition 29

6 Constraint Velocities and Accelerations 30
6.1 State of the Art 30
6.2 Position constraints imply velocity constraints 30
6.3 Acceleration constraints and curvature terms 30
6.4 Five-link constraint derivative hierarchy 31
6.5 Summary 32

7 Closed-Chain Dynamics and Reaction Forces 33
7.1 State of the Art 33
7.2 Force balance for Cartesian rigid bodies 33
7.3 Lagrange multipliers as ideal reactions 33
7.4 Acceleration-level KKT system 34
7.4.1 Symbolic assembly and one constrained solve 35
7.5 Ideal equations and numerical stabilization 35
7.6 Summary 35

8 Consistent Initial Conditions 36
8.1 State of the Art 36
8.2 Position and velocity consistency 36
8.3 Projection of approximate initial data 36
8.3.1 Initialization algorithm 37
8.4 Checks before integration 37
8.5 Summary and transition 37

9 Time Integration and Constraint Drift **38**
9.1 State of the Art . . . 38
9.2 Separate the model from the stepper . . . 38
9.3 Drift control . . . 38
9.3.1 Baumgarte stabilization . . . 38
9.3.2 Projection . . . 39
9.3.3 Other formulations . . . 39
9.4 Time-stepper and tolerances . . . 39
9.5 Algorithmic outline . . . 40
9.5.1 Notebook execution sequence . . . 40
9.6 Summary . . . 40

10 Verification and Validation **41**
10.1 State of the Art . . . 41
10.2 Verification layers . . . 41
10.2.1 Symbolic and equation checks . . . 41
10.2.2 Linear solve checks . . . 42
10.2.3 Trajectory checks . . . 42
10.2.4 Validation evidence . . . 42
10.3 Acceptance checks for the examples . . . 43
10.4 Algorithmic outline . . . 43
10.5 Summary . . . 43

11 Results and Visualization **44**
11.1 State of the Art . . . 44
11.2 Example configuration . . . 44
11.3 Numerical result . . . 44
11.3.1 Figure and animation generation . . . 45
11.4 Sensitivity to tolerances and stabilization . . . 45
11.5 Reproducibility record . . . 46
11.6 Summary . . . 46

12 Limitations and Extensions **47**
12.1 State of the Art . . . 47
12.2 Limitations of the worked model . . . 47
12.3 Possible extensions . . . 47
12.3.1 Spatial motion . . . 47
12.3.2 Flexible bodies and compliant joints . . . 48
12.3.3 Contact, friction, and actuation . . . 48
12.3.4 Robust numerics and model evidence . . . 48
12.4 Summary . . . 48

A Python Implementations **49**
A.1 Double-pendulum acceleration . . . 49
A.2 Five-link constrained acceleration . . . 49
A.3 State derivative and integration . . . 50

General Introduction

Multibody systems are assemblies of bodies whose motion is coupled by joints, force interactions, and geometric relations. When a mechanism contains a closed loop, the bodies cannot move independently: their positions and velocities must remain compatible with loop-closure constraints. This coupling affects modeling, equation derivation, reaction-force recovery, numerical integration, and verification. The purpose of this book is to build a clear route from a physical mechanism to equations that can be simulated and checked.

The intended reader has a working knowledge of introductory mechanics, rigid-body motion, linear algebra, and ordinary differential equations. The book introduces the additional structure needed for constrained systems rather than assuming prior expertise in multibody software. It compares common dynamics formulations, distinguishes coordinate choice from equation and solver choice, derives a small open-chain example, and then applies the ideas to a closed-chain mechanism.

Scope and assumptions

The main case study is a planar loop of five rigid links connected by five ideal revolute pins, with link 4 grounded horizontally. Each body is represented by center-of-mass coordinates (x_i, y_i) and an orientation θ_i. The resulting 15 coordinates are subject to ten scalar pin equations and three fixed-pose equations. At the modeled initial configuration the constraint Jacobian has rank 13, leaving two local degrees of freedom. Rank is configuration-dependent, so this count is not asserted at every possible pose.

The worked model assumes ideal rigid bodies, fixed mass properties, ideal pins, a fixed ground link, and uniform gravity. It omits contact, joint clearance, friction, compliance, flexible deformation, actuation, and experimental calibration. These omissions make the equations easier to inspect; they also limit the model's applicability to real hardware.

An open-chain double pendulum serves a different purpose. It demonstrates how to choose generalized coordinates, derive energies, form Euler–Lagrange equations, extract the mass matrix and remaining terms symbolically, and prepare a first-order initial-value problem. It is explicitly not a closed-chain example. The two examples together show which parts of a workflow come from Lagrangian mechanics and which additional equations arise from loop closure.

How to use the book

Equations, derivations, and language-neutral pseudocode appear in the chapters. The Jupyter notebooks provide executable symbolic derivations, numerical checks, plots, and animation; Python implementation listings are collected in the final appendix. The double-pendulum notebook checks its symbolic decomposition and energy behavior. The five-link

notebook checks the position constraints, their Jacobian and acceleration bias, the KKT solve, and trajectory closure.

Each chapter includes a focused State of the Art section. The introductory synthesis that follows gives a book-wide orientation and states its coverage boundary. These sections are selective and tied to the sources cited; they should not be read as exhaustive systematic reviews.

Roadmap

Chapter 1 compares Newton–Euler, Lagrange, virtual-work, Kane, and Hamiltonian formulations. Chapters 2 and 3 define the mechanism, coordinate system, and rigid-body kinematics. Chapter 4 derives and simulates the double pendulum. Chapters 5–8 formulate loop constraints, differentiate them, assemble the constrained dynamics, and construct consistent initial conditions. Chapters 9–11 discuss integration and drift, verification and validation, and the five-link results. Chapter 12 records the assumptions' limits and outlines possible extensions. The appendix gathers the reader-facing Python implementations.

General State of the Art

Coverage boundary and search scope

This synthesis combines foundational mechanics and numerical-analysis sources with paired reviews of constraint-enforcement methods, a DAE-focused multibody chapter, and selected studies of projection, stabilization, vehicle dynamics, and flexible-body formulations. The targeted discovery pass used Crossref title and DOI records and publisher-linked abstracts on 2026-10-07, with search terms covering multibody dynamics roots, constraint enforcement, DAE methods, stabilization, projection, constraint drift, vehicle-system numerical methods, and flexible multibody dynamics. Bibliographic metadata were checked against Crossref or publisher records; software-specific claims are checked against official SciPy documentation. This is a scoped narrative review, not a systematic review: it has no exhaustive database protocol or quantitative screening, and does not claim to cover every application area or publication through the search date.

Formulations, coordinates, and model structure

The field draws on analytical mechanics, numerical analysis, and application-specific modeling. Schiehlen's historical perspective traces the development of computer-oriented multibody dynamics, while Shabana gives a broad treatment of rigid and deformable systems [1, 2]. Newton–Euler balances, Lagrange equations, virtual-work principles, Kane's equations, and Hamiltonian methods remain alternative descriptions of the same mechanics. They differ in the variables and intermediate quantities they expose, how they handle ideal reactions, and how naturally they support particular symbolic, recursive, or computational workflows.

For constrained systems, formulation and coordinate choice are tightly coupled but not identical. Minimal or reduced coordinates eliminate dependent variables and can shrink the solve, yet a globally valid parameterization may be difficult to obtain and local parameterizations can become singular. Redundant Cartesian or natural coordinates make body poses and joint geometry explicit but introduce algebraic equations and possibly redundant constraints. The paired reviews by Laulusa and Bauchau classify classical formulations and contemporary constraint-enforcement approaches, emphasizing that computational convenience, accuracy, efficiency, and model structure must be considered together rather than ranking one formulation as universally best [3, 4].

Constraint equations and DAE structure

With redundant coordinates, holonomic equations $\Phi(q) = 0$ produce velocity and acceleration conditions by differentiation. Lagrange multipliers place ideal reaction forces among the unknowns and lead to a saddle-point solve. At position level, the coupled differential

and algebraic equations form a DAE; differentiating constraints to acceleration level and solving for accelerations inside an ODE right-hand side is a different computational route, not a dedicated solution of the original DAE. DAE-specific literature treats differentiation index, consistency, index reduction, and the numerical consequences of these choices; Arnold's multibody-focused chapter connects those issues to practical formulations [5, 6, 7].

Constraint rank and conditioning are structural properties, not just solver details. Redundant rows may leave the physical motion unchanged while making multiplier solutions nonunique or the KKT matrix singular. Rank can also change with configuration, so local mobility and solvability must be assessed at the poses of interest. This is one reason reduced-coordinate and explicit redundant-coordinate models can behave differently near singular configurations even when they represent the same ideal mechanism.

Constraint enforcement and drift control

The constraint-enforcement literature distinguishes several families. Index-reduction and differentiated-constraint methods change the equation form; Baumgarte feedback modifies acceleration constraints with position and velocity error terms; projection methods correct positions, velocities, or both; coordinate partitioning or null-space methods eliminate dependent coordinates; penalty and augmented-Lagrangian methods trade exact algebraic closure for modified force or multiplier formulations. Reviews compare these approaches by stability, constraint error, computational cost, conditioning, and suitability to the discretization [8, 3, 4].

The distinctions matter. Stabilization controls drift through feedback and introduces gains whose values affect transient error and numerical stiffness. Projection or constraint-elimination methods seek exact or near-machine-precision consistency, but projection can alter energy or order of accuracy unless designed together with the integrator. Energy-related effects of velocity projection have been analyzed explicitly, and drift-elimination methods provide alternatives to simply increasing feedback gains [9, 10]. The literature therefore supports a menu of tradeoffs, not a single method that dominates for all mechanisms, integration intervals, and quantities of interest.

Time integration and application domains

Time integration is a separate choice from constraint enforcement. General-purpose adaptive ODE methods estimate local error, but an acceleration-level stabilized system may become stiff when correction gains introduce fast time scales. DAE-aware methods instead preserve algebraic variables and have their own consistency and index requirements. For long-time mechanics, energy–momentum and Lie-group methods seek to respect geometric or invariant structure, while generalized-α and related methods may deliberately damp unresolved high-frequency response. These properties do not remove the need to state how constraints are imposed. Numerical-analysis work includes error analysis of generalized-α Lie-group integrators for constrained mechanics; DAE-oriented treatments address the distinct algebraic formulation [11, 12, 7].

The balance also varies by application. Vehicle dynamics has developed specialized numerical methods for large, often stiff or contact-rich models and for real-time simulation; its review literature should not be assumed to represent every branch of multibody research [13]. Flexible multibody dynamics adds deformation coordinates, constitutive models, and elastic energy scales; Shabana's review documents how this branch developed alongside rigid-body methods [14]. Contact and friction introduce unilateral or nonsmooth conditions

and demand methods beyond the smooth equality-constrained example in this book. These application areas broaden the field but also make a single global comparison impractical.

For the software demonstration in this book, SciPy's `solve_ivp` uses RK45 by default; Radau and BDF are implicit alternatives for stiff ODEs. Its output sampling parameter specifies stored times rather than forcing internal solver steps. Documentation version, tolerances, maximum step, solver status, and residual diagnostics should therefore accompany computational results [15].

Verification, validation, and reproducibility

Verification asks whether the equations and implementation have been solved as intended; validation asks whether the model represents reality adequately for its intended use. They need different evidence. Symbolic identities, independent implementations, residuals, benchmark comparisons, and refinement studies support verification. Measurements, quantified uncertainty, and application-specific comparison criteria are needed for empirical validation. Reproducibility requires enough artifacts and computational context to repeat the analysis [16, 17]. The five-link example supplies internal and numerical verification evidence, but no experimental data; low constraint error is not physical validation.

Synthesis and open challenges

Across the surveyed sources, several persistent challenges recur: selecting coordinates that remain useful near singular poses; handling redundant constraints without loss of solvability; balancing closure accuracy against energy behavior, stability, and computational cost; preparing consistent initial data; and making constrained linear algebra scalable. Application-specific developments add real-time requirements, contact and friction, flexible-body constitutive detail, and parameter identification. Verification must also remain distinct from model validation: a numerically precise solution can still represent the wrong physics.

This book adopts one transparent path through that landscape rather than claiming to settle it: minimal coordinates introduce energy-based derivation in the double pendulum, then redundant Cartesian coordinates and multipliers make loop closure explicit in the five-link example. Initial projection, Baumgarte feedback, an adaptive ODE integrator, and trajectory-wide residual checks are treated as separate decisions. The examples are pedagogical and internally checked; they do not establish universal method superiority or experimental validity.

Chapter 1

Dynamics Formulations

1.1 State of the Art

Classical analytical mechanics and multibody dynamics use several equivalent formulations, each organizing coordinates, forces, and constraints differently [18, 2]. This focused review uses these foundational texts to distinguish the formulations needed for this book; it does not compare current software packages or claim an exhaustive review of recent method-performance studies. Bibliographic scope checked 2026-10-07.

1.2 The same mechanics, different formulations

The governing physical system does not change when its equations are written using Newton–Euler balances, Lagrange's equations, virtual work, Kane's method, or Hamiltonian variables. What changes is the choice of variables and how applied forces, ideal reactions, and constraints enter the equations. This distinction matters especially in a closed chain: a convenient formulation can expose reaction forces directly, eliminate them through admissible virtual motion, or retain them as multipliers.

1.2.1 Newton–Euler equations

For each rigid body, Newton–Euler mechanics applies the balance of linear momentum and angular momentum. In planar motion these include two translational equations and one moment equation per body. Joint forces and moments appear explicitly as unknown reactions, so free-body diagrams and load paths are easy to interpret. The cost is that internal reactions must be introduced consistently for every joint; in a closed loop, the resulting body equations and reaction unknowns are coupled around the loop.

This formulation is attractive when individual joint loads are a primary output or when the free-body structure is simple. It becomes algebraically cumbersome when many interconnected bodies create numerous reaction components and loop equations.

1.2.2 Lagrangian equations

Choose generalized coordinates $\boldsymbol{q}$, form kinetic and potential energies T and V, and define $L = T - V$. For each coordinate, the Euler–Lagrange equation is

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i, \tag{1.1}$$

where Q_i is the generalized applied force associated with q_i. With independent coordinates, ideal constraint reactions can often be eliminated because they do no virtual work in admissible motions. With redundant coordinates, constraints can instead be retained explicitly and their reactions represented by Lagrange multipliers. The energy-based procedure is systematic, but deriving a useful coordinate set and evaluating symbolic expressions still require care.

Chapter 4 uses this formulation for the open-chain double pendulum, where two angles describe the configuration directly. The five-link closed-chain model instead uses Cartesian body coordinates and explicit pin constraints; its multipliers are retained in the constrained equations.

1.2.3 Virtual work and d'Alembert's principle

The principle of virtual work states that ideal constraint forces do no work for virtual displacements compatible with the constraints. d'Alembert's principle applies the same idea to dynamics by including inertial forces in the virtual-work balance. With independent admissible virtual displacements, ideal reactions can be removed from the equations. This gives a useful conceptual route to reduced equations and connects force balance with the variational formulation, but constructing admissible motions may be difficult for complicated closed loops.

1.2.4 Kane's method

Kane's method organizes equations using generalized speeds and partial velocities. It can produce compact equations for systems where generalized speeds are convenient or do not coincide simply with coordinate derivatives. Its bookkeeping can be effective for larger multibody systems, although readers must learn its speed and partial-velocity notation and carefully relate generalized speeds to configuration coordinates.

1.2.5 Hamiltonian formulation

Hamiltonian mechanics uses generalized positions and conjugate momenta, giving first-order evolution equations in phase space. It highlights canonical structure and is useful in geometric mechanics and long-term energy analysis. Constraints still require explicit treatment or reduction, and the position–momentum formulation is not automatically simpler for the constrained rigid-body examples in this book.

1.3 How to choose a formulation

The following comparison is qualitative: the best choice depends on the desired outputs, coordinate structure, and constraint treatment.

Method	Starting principle	Typical equations or unknowns	Strength	Tradeoff
Newton–Euler	Linear and angular momentum balances	Accelerations and joint-reaction forces	Reactions and load paths are explicit	Loop reactions are coupled
Lagrange	Kinetic and potential energy	Generalized-coordinate equations; multipliers if redundant	Systematic and symbolic-friendly	Coordinates and constraints matter
Virtual work	d'Alembert balance over admissible variations	Projected force balance; ideal reactions cancel	Avoids solving for ideal reactions	Admissible variations can be difficult
Kane	Generalized speeds and partial velocities	Generalized active and inertial forces	Can produce compact equations	Requires specialized bookkeeping
Hamiltonian	Legendre transform to conjugate momenta	First-order position–momentum equations	Exposes phase-space structure	Constraints need extra treatment

1.4 One double pendulum, five formulations

To compare the methods directly, use the same unforced open-chain model as Chapter 4: point masses m_1 and m_2 joined by massless rigid rods of lengths l_1 and l_2, with the first pivot fixed and gravity acting downward. The rods and pins are ideal, and no actuator torque or other applied force is included in this comparison. This restriction matters: a prescribed joint torque requires an actuator model and corresponding moment balances for the links, beyond the axial-force model used here. Use absolute angles θ_1 and θ_2, positive clockwise from the downward vertical, and let world y point upward. Define $\boldsymbol{q} = [\theta_1, \theta_2]^T$ and $\delta = \theta_1 - \theta_2$. Figure 1.1 separates these generalized coordinates from the fixed model parameters. Its numerical values illustrate the simulation example; all derivations below retain the parameters symbolically. The mass positions are

$$\boldsymbol{r}_1 = l_1 \begin{bmatrix} \sin\theta_1 \\ -\cos\theta_1 \end{bmatrix}, \qquad \boldsymbol{r}_2 = \boldsymbol{r}_1 + l_2 \begin{bmatrix} \sin\theta_2 \\ -\cos\theta_2 \end{bmatrix}. \tag{1.2}$$

Let $\boldsymbol{e}_i = [\sin\theta_i, -\cos\theta_i]^T$ point from the upper end of rod i toward its lower end, and let $\boldsymbol{n}_i = [\cos\theta_i, \sin\theta_i]^T$. These are orthogonal unit vectors, with $\partial \boldsymbol{e}_i/\partial\theta_i = \boldsymbol{n}_i$ and $\partial \boldsymbol{n}_i/\partial\theta_i = -\boldsymbol{e}_i$. Thus $\dot{\boldsymbol{e}}_i = \boldsymbol{n}_i\dot{\theta}_i$ and $\dot{\boldsymbol{n}}_i = -\boldsymbol{e}_i\dot{\theta}_i$. Differentiating the positions once gives $\dot{\boldsymbol{r}}_1 = l_1\boldsymbol{n}_1\dot{\theta}_1$ and $\dot{\boldsymbol{r}}_2 = \dot{\boldsymbol{r}}_1 + l_2\boldsymbol{n}_2\dot{\theta}_2$; differentiating again gives

$$\begin{aligned} \ddot{\boldsymbol{r}}_1 &= l_1\left(\boldsymbol{n}_1\ddot{\theta}_1 - \boldsymbol{e}_1\dot{\theta}_1^2\right), \\ \ddot{\boldsymbol{r}}_2 &= \ddot{\boldsymbol{r}}_1 + l_2\left(\boldsymbol{n}_2\ddot{\theta}_2 - \boldsymbol{e}_2\dot{\theta}_2^2\right). \end{aligned} \tag{1.3}$$

Now draw a free-body diagram for each point mass. Define R_i as a signed axial force, positive in tension. Rod 1 pulls mass 1 toward its upper pin, in direction $-\boldsymbol{e}_1$. Rod 2 pulls mass 1 toward mass 2, in direction $+\boldsymbol{e}_2$, and pulls mass 2 toward mass 1, in direction

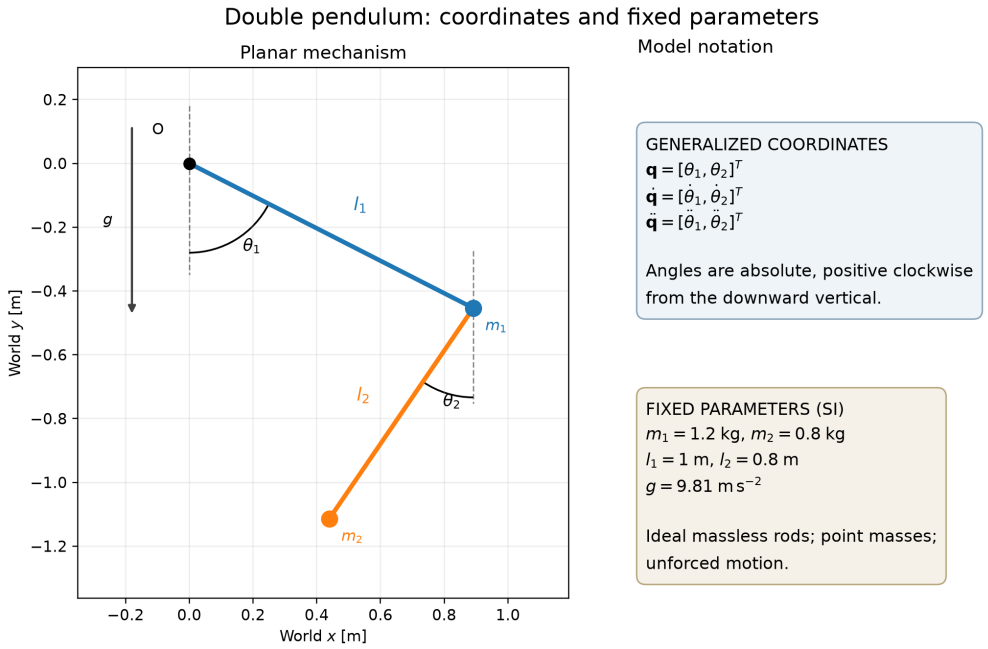


Figure 1.1: Double-pendulum geometry and notation. The generalized coordinates are θ_1 and θ_2; m_1, m_2, l_1, l_2, and g are fixed model parameters. The displayed SI values are the numerical example values, not substitutions used in the symbolic derivations.

$-\boldsymbol{e}_2$; these are the equal-and-opposite forces at their shared pin. Gravity is $-m_i g\boldsymbol{j}$, where $\boldsymbol{j} = [0, 1]^T$ points upward. Applying $\sum \boldsymbol{F} = m\boldsymbol{a}$ separately to each mass now gives

$$\begin{aligned} m_1\ddot{\boldsymbol{r}}_1 &= -R_1\boldsymbol{e}_1 + R_2\boldsymbol{e}_2 - m_1 g\boldsymbol{j}, \\ m_2\ddot{\boldsymbol{r}}_2 &= -R_2\boldsymbol{e}_2 - m_2 g\boldsymbol{j}. \end{aligned} \tag{1.4}$$

These signs can be checked directly: the $R_2\boldsymbol{e}_2$ force on mass 1 is opposite the $-R_2\boldsymbol{e}_2$ force on mass 2, and gravity has negative vertical component. Substitution of the geometric accelerations in (1.3) produces four scalar equations in $\ddot{\theta}_1, \ddot{\theta}_2, R_1, R_2$. Newton–Euler retains all four unknowns; a reduced-coordinate method eliminates the reactions and later recovers them if needed. The next subsections spell out these different operations rather than treating the balances as unexplained results.

1.4.1 Newton–Euler: retain the rod reactions

Substitute (1.3) into (1.4). Move all forces in each balance to the left and define

$$\boldsymbol{f}_1 = m_1\ddot{\boldsymbol{r}}_1 + R_1\boldsymbol{e}_1 - R_2\boldsymbol{e}_2 + m_1 g\boldsymbol{j}, \qquad \boldsymbol{f}_2 = m_2\ddot{\boldsymbol{r}}_2 + R_2\boldsymbol{e}_2 + m_2 g\boldsymbol{j}. \tag{1.5}$$

Both residuals are zero by Newton's second law. Differentiate the geometry and take the needed dot products:

$$\begin{aligned} \boldsymbol{n}_1 \cdot \ddot{\boldsymbol{r}}_1 &= l_1\ddot{\theta}_1, \\ \boldsymbol{n}_1 \cdot \ddot{\boldsymbol{r}}_2 &= l_1\ddot{\theta}_1 + l_2\cos\delta\,\ddot{\theta}_2 + l_2\sin\delta\,\dot{\theta}_2^2, \\ \boldsymbol{n}_2 \cdot \ddot{\boldsymbol{r}}_2 &= l_1\cos\delta\,\ddot{\theta}_1 - l_1\sin\delta\,\dot{\theta}_1^2 + l_2\ddot{\theta}_2. \end{aligned} \tag{1.6}$$

To eliminate internal forces, project $\boldsymbol{f}_1 + \boldsymbol{f}_2$ along $l_1\boldsymbol{n}_1$; rod-2 forces cancel by action and reaction, and rod 1 is perpendicular to $\boldsymbol{n}_1$. Project $\boldsymbol{f}_2$ along $l_2\boldsymbol{n}_2$; rod 2 is perpendicular

to $\boldsymbol{n}_2$. Inserting (1.6) and $\boldsymbol{n}_i \cdot \boldsymbol{j} = \sin\theta_i$ gives

$$\begin{aligned}(m_1+m_2)l_1^2\ddot{\theta}_1 + m_2 l_1 l_2 \cos\delta\,\ddot{\theta}_2 + m_2 l_1 l_2 \sin\delta\,\dot{\theta}_2^2 + (m_1+m_2)gl_1\sin\theta_1 = 0,\\ m_2 l_1 l_2 \cos\delta\,\ddot{\theta}_1 + m_2 l_2^2\ddot{\theta}_2 - m_2 l_1 l_2 \sin\delta\,\dot{\theta}_1^2 + m_2 g l_2 \sin\theta_2 = 0.\end{aligned} \tag{1.7}$$

These are the two angular equations; they are the unforced specialization of the mass-matrix equation derived below. After solving them for $\ddot{\theta}_1, \ddot{\theta}_2$, recover R_2 by projecting the second-body balance along $\boldsymbol{e}_2$, and recover R_1 by projecting the first-body balance along $\boldsymbol{e}_1$:

$$\begin{aligned}R_2 &= m_2\left(l_1\sin\delta\,\ddot{\theta}_1 + l_1\cos\delta\,\dot{\theta}_1^2 + l_2\dot{\theta}_2^2 + g\cos\theta_2\right),\\ R_1 &= m_1 l_1\dot{\theta}_1^2 + R_2\cos\delta + m_1 g\cos\theta_1.\end{aligned} \tag{1.8}$$

The method's final body-level equations are the original balances, now with the recovered accelerations and reactions substituted:

$$\begin{aligned}m_1\ddot{\boldsymbol{r}}_1 &= -R_1\boldsymbol{e}_1 + R_2\boldsymbol{e}_2 - m_1 g\boldsymbol{j},\\ m_2\ddot{\boldsymbol{r}}_2 &= -R_2\boldsymbol{e}_2 - m_2 g\boldsymbol{j}.\end{aligned} \tag{1.9}$$

1.4.2 Lagrange: derive directly in angles

The two angles satisfy the rod geometry automatically, so no rod reactions are needed as unknowns. The velocities from (1.3) before differentiation are $\dot{\boldsymbol{r}}_1 = l_1\boldsymbol{n}_1\dot{\theta}_1$ and $\dot{\boldsymbol{r}}_2 = l_1\boldsymbol{n}_1\dot{\theta}_1 + l_2\boldsymbol{n}_2\dot{\theta}_2$. Since $\boldsymbol{n}_i\cdot\boldsymbol{n}_i = 1$ and $\boldsymbol{n}_1\cdot\boldsymbol{n}_2 = \cos\delta$, their squared speeds are

$$\begin{aligned}\dot{\boldsymbol{r}}_1\cdot\dot{\boldsymbol{r}}_1 &= l_1^2\dot{\theta}_1^2,\\ \dot{\boldsymbol{r}}_2\cdot\dot{\boldsymbol{r}}_2 &= l_1^2\dot{\theta}_1^2 + l_2^2\dot{\theta}_2^2 + 2l_1 l_2\cos\delta\,\dot{\theta}_1\dot{\theta}_2.\end{aligned} \tag{1.10}$$

Substitute these into $T = \frac{1}{2}m_1\dot{\boldsymbol{r}}_1^2 + \frac{1}{2}m_2\dot{\boldsymbol{r}}_2^2$. The heights from the position definitions are $y_1 = -l_1\cos\theta_1$ and $y_2 = -l_1\cos\theta_1 - l_2\cos\theta_2$; use $V = m_1 g y_1 + m_2 g y_2$. The resulting energies are

$$\begin{aligned}T &= \frac{1}{2}(m_1+m_2)l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2 l_2^2\dot{\theta}_2^2 + m_2 l_1 l_2\dot{\theta}_1\dot{\theta}_2\cos\delta,\\ V &= -(m_1+m_2)gl_1\cos\theta_1 - m_2 g l_2\cos\theta_2.\end{aligned} \tag{1.11}$$

Set $L = T - V$. Since $\delta = \theta_1 - \theta_2$, its time derivative is $\dot{\delta} = \dot{\theta}_1 - \dot{\theta}_2$. The derivatives in the Euler–Lagrange equations are

$$\begin{aligned}\frac{d}{dt}\frac{\partial L}{\partial\dot{\theta}_1} &= (m_1+m_2)l_1^2\ddot{\theta}_1 + m_2 l_1 l_2\left[\ddot{\theta}_2\cos\delta - \dot{\theta}_2\sin\delta(\dot{\theta}_1 - \dot{\theta}_2)\right],\\ \frac{\partial L}{\partial\theta_1} &= -m_2 l_1 l_2\dot{\theta}_1\dot{\theta}_2\sin\delta - (m_1+m_2)gl_1\sin\theta_1,\\ \frac{d}{dt}\frac{\partial L}{\partial\dot{\theta}_2} &= m_2 l_2^2\ddot{\theta}_2 + m_2 l_1 l_2\left[\ddot{\theta}_1\cos\delta - \dot{\theta}_1\sin\delta(\dot{\theta}_1 - \dot{\theta}_2)\right],\\ \frac{\partial L}{\partial\theta_2} &= m_2 l_1 l_2\dot{\theta}_1\dot{\theta}_2\sin\delta - m_2 g l_2\sin\theta_2.\end{aligned} \tag{1.12}$$

For coordinate i, subtract $\partial L/\partial\theta_i$ from $d(\partial L/\partial\dot{\theta}_i)/dt$. In the first equation the mixed product $\dot{\theta}_1\dot{\theta}_2\sin\delta$ cancels, leaving the $\dot{\theta}_2^2$ term; in the second it cancels leaving $-\dot{\theta}_1^2$. Thus the two Euler–Lagrange equations are

$$\begin{aligned}(m_1+m_2)l_1^2\ddot{\theta}_1 + m_2 l_1 l_2\cos\delta\,\ddot{\theta}_2 + m_2 l_1 l_2\sin\delta\,\dot{\theta}_2^2 + (m_1+m_2)gl_1\sin\theta_1 = 0,\\ m_2 l_1 l_2\cos\delta\,\ddot{\theta}_1 + m_2 l_2^2\ddot{\theta}_2 - m_2 l_1 l_2\sin\delta\,\dot{\theta}_1^2 + m_2 g l_2\sin\theta_2 = 0.\end{aligned} \tag{1.13}$$

Collecting the acceleration coefficients defines the mass matrix and the remaining terms define the bias vector:

$$M = \begin{bmatrix} (m_1+m_2)l_1^2 & m_2 l_1 l_2 \cos\delta \\ m_2 l_1 l_2 \cos\delta & m_2 l_2^2 \end{bmatrix}, \qquad \boldsymbol{h} = \begin{bmatrix} m_2 l_1 l_2 \sin\delta\, \dot\theta_2^2 + (m_1+m_2) g l_1 \sin\theta_1 \\ -m_2 l_1 l_2 \sin\delta\, \dot\theta_1^2 + m_2 g l_2 \sin\theta_2 \end{bmatrix}. \tag{1.14}$$

For this unforced example the reduced angular equations are

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{0}. \tag{1.15}$$

Solve for $\ddot{\boldsymbol{q}}$ and substitute into (1.3). The lower-link axial balance gives R_2; the upper-link axial balance then gives R_1:

$$\begin{aligned} R_2 &= m_2 \left(l_1 \sin\delta\, \ddot\theta_1 + l_1 \cos\delta\, \dot\theta_1^2 + l_2 \dot\theta_2^2 + g\cos\theta_2 \right), \\ R_1 &= m_1 l_1 \dot\theta_1^2 + R_2 \cos\delta + m_1 g \cos\theta_1. \end{aligned} \tag{1.16}$$

Putting these reactions back into Newton's second-law equations recovers the body balances:

$$\begin{aligned} m_1 \ddot{\boldsymbol{r}}_1 &= -R_1 \boldsymbol{e}_1 + R_2 \boldsymbol{e}_2 - m_1 g \boldsymbol{j}, \\ m_2 \ddot{\boldsymbol{r}}_2 &= -R_2 \boldsymbol{e}_2 - m_2 g \boldsymbol{j}. \end{aligned} \tag{1.17}$$

1.4.3 Virtual work and d'Alembert: project onto admissible motion

For virtual angle changes $\delta\theta_1, \delta\theta_2$, the induced displacements are $\delta\boldsymbol{r}_1 = l_1 \boldsymbol{n}_1 \delta\theta_1$ and $\delta\boldsymbol{r}_2 = l_1 \boldsymbol{n}_1 \delta\theta_1 + l_2 \boldsymbol{n}_2 \delta\theta_2$. Each ideal axial rod force is perpendicular to the compatible displacement of its endpoint, so its virtual work is zero. d'Alembert's principle says that the sum of applied-force and inertial-force virtual work is zero. Equating the independent coefficients of $\delta\theta_1$ and $\delta\theta_2$ gives

$$\begin{aligned} l_1 \boldsymbol{n}_1 \cdot (m_1 \ddot{\boldsymbol{r}}_1 + m_2 \ddot{\boldsymbol{r}}_2 + (m_1+m_2) g \boldsymbol{j}) &= 0, \\ l_2 \boldsymbol{n}_2 \cdot (m_2 \ddot{\boldsymbol{r}}_2 + m_2 g \boldsymbol{j}) &= 0. \end{aligned} \tag{1.18}$$

Substitute the geometric acceleration projections in (1.6). Before collecting coefficients, the two scalar balances read

$$\begin{aligned} l_1 \big[(m_1+m_2) l_1 \ddot\theta_1 + m_2 l_2 \cos\delta\, \ddot\theta_2 + m_2 l_2 \sin\delta\, \dot\theta_2^2 + (m_1+m_2) g \sin\theta_1 \big] &= 0, \\ l_2 m_2 \big[l_1 \cos\delta\, \ddot\theta_1 - l_1 \sin\delta\, \dot\theta_1^2 + l_2 \ddot\theta_2 + g \sin\theta_2 \big] &= 0. \end{aligned} \tag{1.19}$$

Divide the first line by l_1 and the second by $l_2 m_2$, then collect terms. The resulting equations are

$$\begin{aligned} &(m_1+m_2) l_1^2 \ddot\theta_1 + m_2 l_1 l_2 \cos\delta\, \ddot\theta_2 + m_2 l_1 l_2 \sin\delta\, \dot\theta_2^2 + (m_1+m_2) g l_1 \sin\theta_1 = 0, \\ &m_2 l_1 l_2 \cos\delta\, \ddot\theta_1 + m_2 l_2^2 \ddot\theta_2 - m_2 l_1 l_2 \sin\delta\, \dot\theta_1^2 + m_2 g l_2 \sin\theta_2 = 0. \end{aligned} \tag{1.20}$$

These match the Lagrange equations because the virtual displacements are exactly the coordinate variations. Once they determine the angular acceleration, recover reactions using (1.16); substituting these into the two body balances gives

$$\begin{aligned} m_1 \ddot{\boldsymbol{r}}_1 &= -R_1 \boldsymbol{e}_1 + R_2 \boldsymbol{e}_2 - m_1 g \boldsymbol{j}, \\ m_2 \ddot{\boldsymbol{r}}_2 &= -R_2 \boldsymbol{e}_2 - m_2 g \boldsymbol{j}. \end{aligned} \tag{1.21}$$

1.4.4 Kane: use generalized speeds and partial velocities

Choose generalized speeds $u_1 = \dot{\theta}_1$ and $u_2 = \dot{\theta}_2$. Differentiating the point velocities with respect to the speeds gives the partial velocities

$$\boldsymbol{v}_{11} = l_1\boldsymbol{n}_1, \qquad \boldsymbol{v}_{21} = l_1\boldsymbol{n}_1, \qquad \boldsymbol{v}_{22} = l_2\boldsymbol{n}_2, \qquad \boldsymbol{v}_{12} = \boldsymbol{0}. \tag{1.22}$$

Kane's generalized active force is $F_k = \sum_i \boldsymbol{v}_{ik} \cdot \boldsymbol{F}_i$ and generalized inertia force is $F_k^* = -\sum_i m_i \boldsymbol{v}_{ik} \cdot \ddot{\boldsymbol{r}}_i$. Here gravity gives $F_1 = -(m_1 + m_2)gl_1 \sin\theta_1$ and $F_2 = -m_2 g l_2 \sin\theta_2$. Projecting the accelerations in (1.3) onto the partial velocities gives

$$\begin{aligned} F_1^* &= -\left[(m_1 + m_2)l_1^2\ddot{\theta}_1 + m_2 l_1 l_2 \cos\delta\,\ddot{\theta}_2 + m_2 l_1 l_2 \sin\delta\,\dot{\theta}_2^2\right], \\ F_2^* &= -\left[m_2 l_1 l_2 \cos\delta\,\ddot{\theta}_1 + m_2 l_2^2\ddot{\theta}_2 - m_2 l_1 l_2 \sin\delta\,\dot{\theta}_1^2\right]. \end{aligned} \tag{1.23}$$

Setting $F_k + F_k^* = 0$ and inserting the active forces gives the two equations explicitly:

$$\begin{aligned} &(m_1 + m_2)l_1^2\ddot{\theta}_1 + m_2 l_1 l_2 \cos\delta\,\ddot{\theta}_2 + m_2 l_1 l_2 \sin\delta\,\dot{\theta}_2^2 + (m_1 + m_2)gl_1 \sin\theta_1 = 0, \\ &m_2 l_1 l_2 \cos\delta\,\ddot{\theta}_1 + m_2 l_2^2\ddot{\theta}_2 - m_2 l_1 l_2 \sin\delta\,\dot{\theta}_1^2 + m_2 g l_2 \sin\theta_2 = 0. \end{aligned} \tag{1.24}$$

These match the virtual-work equations term by term. Kane's method likewise eliminates ideal rod reactions from the acceleration solve. After solving for the accelerations, use (1.16) to recover those reactions; the resulting body-level force balances are

$$\begin{aligned} m_1\ddot{\boldsymbol{r}}_1 &= -R_1\boldsymbol{e}_1 + R_2\boldsymbol{e}_2 - m_1 g\boldsymbol{j}, \\ m_2\ddot{\boldsymbol{r}}_2 &= -R_2\boldsymbol{e}_2 - m_2 g\boldsymbol{j}. \end{aligned} \tag{1.25}$$

1.4.5 Hamiltonian: evolve positions and momenta

Starting from the Lagrange result, define conjugate momenta by $\boldsymbol{p} = \partial L/\partial\dot{\boldsymbol{q}} = M(\boldsymbol{q})\dot{\boldsymbol{q}}$. Written componentwise,

$$\begin{aligned} p_1 &= (m_1 + m_2)l_1^2\dot{\theta}_1 + m_2 l_1 l_2 \cos\delta\,\dot{\theta}_2, \\ p_2 &= m_2 l_1 l_2 \cos\delta\,\dot{\theta}_1 + m_2 l_2^2\dot{\theta}_2. \end{aligned} \tag{1.26}$$

The determinant is $D = \det M = m_2 l_1^2 l_2^2(m_1 + m_2 \sin^2\delta) > 0$ for positive masses and lengths, so the Legendre transform is invertible. The Hamiltonian $\mathcal{H} = \boldsymbol{p}^T\dot{\boldsymbol{q}} - L$ is the total mechanical energy expressed in $(\boldsymbol{q}, \boldsymbol{p})$:

$$\mathcal{H}(\boldsymbol{q}, \boldsymbol{p}) = \frac{(m_2 l_2^2)p_1^2 - 2m_2 l_1 l_2 \cos\delta\, p_1 p_2 + (m_1 + m_2)l_1^2 p_2^2}{2D} + V(\boldsymbol{q}). \tag{1.27}$$

Hamilton's canonical equations for this unforced model are

$$\dot{\boldsymbol{q}} = \frac{\partial\mathcal{H}}{\partial\boldsymbol{p}}, \qquad \dot{\boldsymbol{p}} = -\frac{\partial\mathcal{H}}{\partial\boldsymbol{q}}. \tag{1.28}$$

The first canonical equation gives $\dot{\boldsymbol{q}} = M^{-1}\boldsymbol{p}$. In components, this solve is

$$\begin{bmatrix}\dot{\theta}_1 \\ \dot{\theta}_2\end{bmatrix} = \frac{1}{D}\begin{bmatrix} m_2 l_2^2 & -m_2 l_1 l_2 \cos\delta \\ -m_2 l_1 l_2 \cos\delta & (m_1 + m_2)l_1^2\end{bmatrix}\begin{bmatrix}p_1 \\ p_2\end{bmatrix}. \tag{1.29}$$

At fixed $\boldsymbol{p}$, differentiating $\mathcal{H} = \frac{1}{2}\boldsymbol{p}^T M^{-1}\boldsymbol{p} + V$ with respect to q_i uses $\partial M^{-1}/\partial q_i = -M^{-1}M_{,i}M^{-1}$, and therefore gives $\partial\mathcal{H}/\partial q_i = -\frac{1}{2}\dot{\boldsymbol{q}}^T M_{,i}\dot{\boldsymbol{q}} + V_{,i}$. Differentiating $\boldsymbol{p} = M\dot{\boldsymbol{q}}$ in time gives $\dot{\boldsymbol{p}} = M\ddot{\boldsymbol{q}} + \dot{M}\dot{\boldsymbol{q}}$. Substitution into Hamilton's second equation yields

$$M\ddot{\boldsymbol{q}} + \dot{M}\dot{\boldsymbol{q}} - \frac{1}{2}\begin{bmatrix}\dot{\boldsymbol{q}}^T M_{,1}\dot{\boldsymbol{q}} \\ \dot{\boldsymbol{q}}^T M_{,2}\dot{\boldsymbol{q}}\end{bmatrix} + \nabla_{\boldsymbol{q}} V = \boldsymbol{0}, \tag{1.30}$$

where $M_{,i} = \partial M/\partial q_i$. Let $A = m_2 l_1 l_2$. The two coordinate derivatives of the mass matrix are

$$M_{,1} = \begin{bmatrix}0 & -A\sin\delta \\ -A\sin\delta & 0\end{bmatrix}, \qquad M_{,2} = \begin{bmatrix}0 & A\sin\delta \\ A\sin\delta & 0\end{bmatrix}. \tag{1.31}$$

The chain rule gives $\dot{M} = M_{,1}\dot{\theta}_1 + M_{,2}\dot{\theta}_2$. Substitution into the velocity-dependent term yields

$$\dot{M}\dot{\boldsymbol{q}} - \frac{1}{2}\begin{bmatrix}\dot{\boldsymbol{q}}^T M_{,1}\dot{\boldsymbol{q}} \\ \dot{\boldsymbol{q}}^T M_{,2}\dot{\boldsymbol{q}}\end{bmatrix} = \begin{bmatrix}A\sin\delta\,\dot{\theta}_2^2 \\ -A\sin\delta\,\dot{\theta}_1^2\end{bmatrix}. \tag{1.32}$$

The potential gradient is

$$\nabla_{\boldsymbol{q}} V = \begin{bmatrix}(m_1+m_2)gl_1\sin\theta_1 \\ m_2 g l_2 \sin\theta_2\end{bmatrix}. \tag{1.33}$$

These are exactly the velocity and gravity terms in (1.14); the Hamilton equations therefore reduce to (1.13). Solve them for angular accelerations and use (1.16) to recover the axial forces. Substitution in Newton's second-law balances returns

$$\begin{aligned} m_1\ddot{\boldsymbol{r}}_1 &= -R_1\boldsymbol{e}_1 + R_2\boldsymbol{e}_2 - m_1 g\boldsymbol{\jmath}, \\ m_2\ddot{\boldsymbol{r}}_2 &= -R_2\boldsymbol{e}_2 - m_2 g\boldsymbol{\jmath}. \end{aligned} \tag{1.34}$$

Do not confuse a dynamics formulation with other modeling decisions. Minimal versus redundant coordinates are coordinate choices. ODE versus DAE describes the mathematical form presented to the solver. Explicit versus implicit integration is a time-stepping choice. These decisions interact, but they are not interchangeable names for the same choice.

1.5 Formulation used in this book

The book uses two examples to build the reader's intuition. Chapter 4 uses the double pendulum to demonstrate Lagrangian derivation with minimal coordinates and conversion to a first-order ODE. The five-link mechanism then uses redundant Cartesian coordinates, holonomic pin constraints, and Lagrange multipliers, producing a constrained acceleration system. This progression keeps the introductory derivation small while making the additional work required by loop closure explicit.

1.6 Summary

Newton–Euler, Lagrange, virtual-work, Kane, and Hamiltonian formulations describe the same underlying mechanics but expose different variables and tradeoffs. This book uses Lagrange's equations for the double-pendulum tutorial and a multiplier-based Cartesian formulation for the closed-chain case. The following mechanism and kinematics chapters define the model and coordinates before deriving its loop constraints.

Chapter 2

Mechanism and Modeling Assumptions

2.1 State of the Art

Multibody models represent a mechanism as interacting rigid or deformable bodies connected by joints and loads. Modeling choices determine which physical effects the equations can represent and which numerical methods are needed [2]. This focused review uses a foundational multibody text to frame rigid-body and joint idealizations. It does not survey application-specific contact or compliant-joint models. Bibliographic scope checked 2026-10-07.

2.2 Open chains and closed chains

2.3 From equation choice to physical model

Chapter 1 compared ways to express dynamics once a system has been selected. The next decision is the model itself: which bodies exist, how they are connected, what is held fixed, and which physical effects are retained. The book uses two systems for different teaching purposes. The double pendulum is an open-chain example whose two angles satisfy its rod geometry by construction. It lets Chapter 4 develop energy-based equations without loop-closure reactions. The main case study in this chapter is instead a closed chain: five links form a perimeter loop, so the joint geometry must remain closed throughout motion. It is this second model, not the double pendulum, that motivates the constrained-coordinate treatment in the later chapters.

2.4 Five links, five pins, one loop

Number the planar links 1 through 5 around the perimeter. Pin P_{ij} joins links i and j at one point on each link; the complete connection list is

$$\mathcal{E} = \{P_{12}, P_{23}, P_{34}, P_{45}, P_{51}\}. \tag{2.1}$$

Thus links 1–2–3–4–5–1 make one closed loop. Link 4 is attached to ground and held horizontal. Its support is not an additional moving link or one of the five revolute pins: it fixes link 4's planar pose. There is no diagonal pin. Adding one would define a different mechanism and, for the modeled geometry, could remove the intended mobility.

Figure 2.1 shows the labeled body arrangement at initialization and a later simulated configuration. It helps identify the loop and the grounded member; it is a geometry illustration, not evidence by itself that the equations or trajectory are correct.

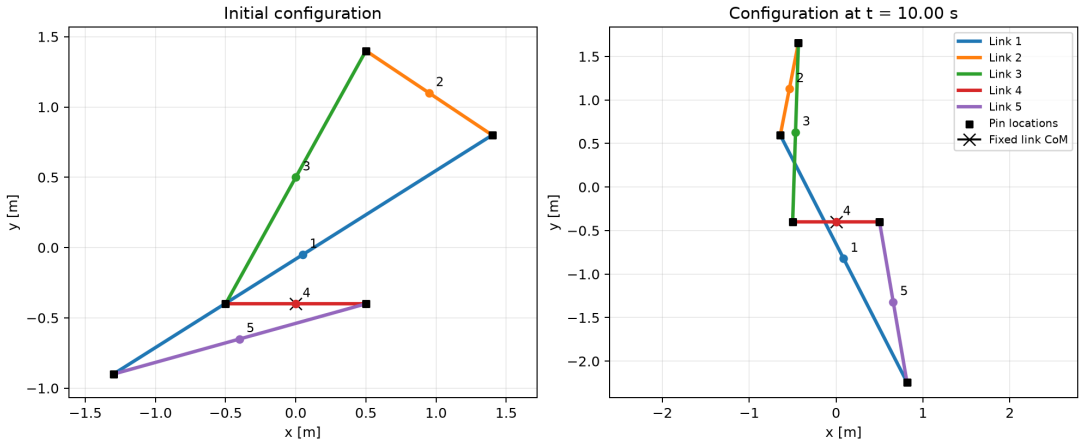


Figure 2.1: Five-link perimeter mechanism at initialization and at $t = 10\,\mathrm{s}$. Colored segments identify links 1–5, black squares mark the revolute pins, and the cross marks the fixed center of mass of grounded link 4. The second panel illustrates a computed configuration; trajectory correctness is assessed with residual and sensitivity checks in later chapters.

The pin inventory makes the loop topology explicit:

oprule Pin	Connected links	Role in the perimeter loop
P_{12}	1 and 2	Connects the first and second links
P_{23}	2 and 3	Connects the second and third links
P_{34}	3 and 4	Connects the third link to ground link 4
P_{45}	4 and 5	Connects ground link 4 to the fifth link
P_{51}	5 and 1	Closes the loop

2.5 Bodies, joints, and loads

Each link is an ideal planar rigid body. Link i has constant mass $m_i > 0$, center of mass at the body-coordinate origin, and planar moment of inertia $I_i > 0$ about that center. A pin transmits force between its connected links and enforces coincidence of the two attachment points, while allowing relative rotation; it transmits no ideal couple. Chapter 3 defines the body coordinates and maps those attachment points into world coordinates, and Chapter 5 turns the five pin connections into explicit loop-closure equations.

Ground fixes the position and orientation of link 4. The reference orientation is chosen as zero, so the grounded link is horizontal with world x; its three pose conditions provide the physical support needed to remove free rigid translation and rotation. The support reactions needed to maintain this pose are internal to the model's constrained force balance, not prescribed external loads.

The baseline external loading is gravity, with world y positive upward. It acts downward on every link's center of mass; no actuator torque or other applied force is included in the five-link baseline. A driven or damped mechanism would need an explicitly defined force or torque model rather than an implicit change to the joint geometry.

2.6 Modeling contract and scope

The equations developed later apply to this specific idealization. Their assumptions and consequences are collected here so the reader can tell which claims belong to the model and which physical effects have been left out.

oprule Model choice	Included representation	Consequence or excluded effect
Motion	Planar translation and rotation	Out-of-plane motion is not represented
Bodies	Rigid links with fixed m_i and I_i	Flexibility and changing mass properties are excluded
Pins	Ideal revolute joints	No friction, backlash, compliance, or transmitted couple
Support	Link 4 fixed horizontally to ground	Support reactions are solved, not prescribed
Loading	Gravity only in the baseline	No actuation, damping, impact, or contact forces

These omissions are scope boundaries, not claims that the effects are negligible in applications. Introducing contact, friction, compliance, flexibility, or actuation changes the force model, the constraints, or both, and requires new verification. The double-pendulum example has a different idealization: it uses point masses and massless rods as an open-chain teaching model, not the five-link rigid-body loop.

2.7 Mobility preview

The model inventory already signals why the five-link equations will be constrained. Five planar bodies have $5 \times 3 = 15$ Cartesian pose coordinates. Each of the five pins requires two scalar point-coincidence conditions, and fixing link 4 adds three pose conditions, for 13 written scalar equations. At the modeled configuration the constraint Jacobian has rank 13, so the local mobility is

$$n_{\mathrm{dof}} = 15 - \mathrm{rank}(J) = 15 - 13 = 2. \tag{2.2}$$

This is an orientation count, not the derivation of the constraints: equation count alone does not establish independence, and rank can change at singular configurations. Chapter 5 constructs the residual and evaluates its rank; Chapter 8 uses the resulting two degrees of freedom when preparing consistent initial motion.

2.8 Summary and transition

The book now has a physical contract for its main case study: five planar rigid links, five perimeter pins, link 4 fixed horizontally, and gravity as the only baseline load. The double pendulum remains a separate open-chain example for comparing dynamics formulations. Chapter 3 defines the Cartesian body coordinates and attachment-point kinematics; Chapter 5 will use them to derive the loop constraints rather than assuming closure from a diagram.

Chapter 3

Coordinates and Rigid-Body Kinematics

3.1 State of the Art

Closed-chain mechanisms can be described using a reduced set of independent coordinates or redundant coordinates together with explicit constraints. Cartesian body coordinates represent body poses directly, while reduced coordinates can produce smaller equations but may be difficult to construct globally and may encounter singularities. This focused review uses the foundational multibody treatment in Shabana; it does not survey recent algorithmic comparisons of coordinate parameterizations. Bibliographic scope checked 2026-10-07 [2].

3.2 Cartesian coordinates for planar bodies

A planar rigid body has two translational coordinates for its center of mass and one orientation coordinate. For body i, define

$$\boldsymbol{q}_i = [x_i, y_i, \theta_i]^T, \qquad \boldsymbol{q} = [\boldsymbol{q}_1^T, \ldots, \boldsymbol{q}_n^T]^T. \tag{3.1}$$

The corresponding velocity and acceleration vectors use the same ordering. For the five-link example, $n = 5$ and $\boldsymbol{q} \in \mathbb{R}^{15}$. The mass matrix for center-of-mass coordinates is block diagonal by body, with each planar block $\mathrm{diag}(m_i, m_i, I_i)$; it is constant for rigid bodies with fixed mass properties.

The Cartesian coordinates are redundant for a mechanism whose motion has fewer degrees of freedom than coordinates. This is useful in a closed chain because every body pose is represented directly and the joint geometry can be imposed with equations. Redundancy is handled later using $\Phi(\boldsymbol{q}) = \boldsymbol{0}$ and its Jacobian.

3.3 Reference frames and planar rotation

World coordinates use x to the right and y upward. Body orientation θ_i is measured counterclockwise from the world x axis. The rotation taking a body-frame vector to world coordinates is

$$A(\theta_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix}. \tag{3.2}$$

This body-orientation convention differs from Chapter 4's pendulum angles, which are measured clockwise from the downward vertical. Define each convention where it is used; do not reuse a symbol with a changed geometric meaning without stating the change.

3.4 World position and velocity of an attachment point

Let $\boldsymbol{s}_i = [s_{ix}, s_{iy}]^T$ be a point fixed in body i, expressed in body coordinates. Its world position is

$$\boldsymbol{r}_i(\boldsymbol{q}_i) = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + A(\theta_i)\boldsymbol{s}_i. \tag{3.3}$$

Define the planar quarter-turn matrix

$$S = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \tag{3.4}$$

Since $\partial A/\partial\theta_i = SA$, differentiating (3.3) gives

$$\dot{\boldsymbol{r}}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} + \dot{\theta}_i SA(\theta_i)\boldsymbol{s}_i. \tag{3.5}$$

Thus the point's Jacobian block with respect to its body's coordinates is

$$\frac{\partial \boldsymbol{r}_i}{\partial \boldsymbol{q}_i} = \begin{bmatrix} I_2 & SA(\theta_i)\boldsymbol{s}_i \end{bmatrix}. \tag{3.6}$$

This block is the kinematic building unit for the pin constraints in Chapter 5. For a pin between bodies a and b, the two world attachment points are equated; the corresponding Jacobian blocks enter with opposite signs.

3.5 Double pendulum: forward and inverse kinematics

Use the angle convention of Chapter 4: θ_1 and θ_2 are absolute angles measured clockwise from downward vertical, while world y points upward. These are open-chain angles, not the Cartesian body orientations measured from world x above. With fixed pivot $O = (0, 0)$, let C be the inter-mass joint and T the tip mass position. Forward kinematics maps the two angles to these points:

$$\begin{aligned} C(\theta_1) &= \begin{bmatrix} l_1 \sin\theta_1 \\ -l_1 \cos\theta_1 \end{bmatrix}, \\ T(\theta_1, \theta_2) &= C(\theta_1) + \begin{bmatrix} l_2 \sin\theta_2 \\ -l_2 \cos\theta_2 \end{bmatrix}. \end{aligned} \tag{3.7}$$

Differentiating with respect to the coordinates gives the tip Jacobian and velocity relation

$$J_T(\boldsymbol{q}) = \frac{\partial T}{\partial \boldsymbol{q}} = \begin{bmatrix} l_1 \cos\theta_1 & l_2 \cos\theta_2 \\ l_1 \sin\theta_1 & l_2 \sin\theta_2 \end{bmatrix}, \qquad \dot{T} = J_T \dot{\boldsymbol{q}}, \qquad \det J_T = l_1 l_2 \sin(\theta_2 - \theta_1). \tag{3.8}$$

The determinant vanishes when the two rods are collinear; at those extended or folded poses, an instantaneous tip velocity cannot be produced uniquely by arbitrary angle rates.

If both mass positions are measured, inverse kinematics is direct. For measured $C = (x_1, y_1)$ and $T = (x_2, y_2)$, the model requires $\|C\| = l_1$ and $\|T - C\| = l_2$, and then

$$\theta_1 = \operatorname{atan2}(x_1, -y_1), \qquad \theta_2 = \operatorname{atan2}(x_2 - x_1, -(y_2 - y_1)). \tag{3.9}$$

Here $\operatorname{atan2}(y, x)$ denotes the quadrant-aware angle whose cosine and sine are proportional to (x, y); the swapped arguments above follow from $x = l\sin\theta$, $y = -l\cos\theta$.

Often only a desired tip $T = (x, y)$ is given. Set $d = \sqrt{x^2 + y^2}$ and define the unit vector $\boldsymbol{u} = T/d$ from the pivot to the target, with perpendicular $\boldsymbol{n} = [-u_y, u_x]^T$. The first joint C lies on both circles: radius l_1 about O and radius l_2 about T. Resolve C into components parallel and perpendicular to $\boldsymbol{u}$:

$$a = \frac{l_1^2 - l_2^2 + d^2}{2d}, \qquad h = \sqrt{l_1^2 - a^2}, \qquad C = a\boldsymbol{u} + \sigma h\boldsymbol{n}, \quad \sigma \in \{-1, +1\}. \tag{3.10}$$

Indeed $C \cdot \boldsymbol{u} = a$ and $\|C\|^2 = a^2 + h^2 = l_1^2$; also $\|T - C\|^2 = (d - a)^2 + h^2 = l_2^2$. The two signs describe the two elbow configurations. For $d > 0$, a real solution exists only when $|l_1 - l_2| \le d \le l_1 + l_2$; equality is a collinear singular configuration where the two branches meet. If $d = 0$ and $l_1 = l_2$, the target is the pivot and every first-link direction is possible, so the unit-vector formula is undefined and the inverse is non-unique. For either nondegenerate intersection, recover

$$\theta_1 = \operatorname{atan2}(C_x, -C_y), \qquad \theta_2 = \operatorname{atan2}(x - C_x, -(y - C_y)). \tag{3.11}$$

3.6 Five-link loop: Cartesian and reduced kinematics

For the closed-chain model, the book's dynamics coordinates remain the redundant Cartesian body poses $\boldsymbol{q}_i = [x_i, y_i, \theta_i]^T$ from (3.1). The forward map for any body-fixed pin $\boldsymbol{s}_i$ is (3.3); its velocity and Jacobian are (3.5)–(3.6). For pin P_{ab}, denote the attachment coordinates on the two bodies by $\boldsymbol{s}_a^{ab}$ and $\boldsymbol{s}_b^{ab}$. Its world gap is

$$\boldsymbol{\Phi}_{ab}(\boldsymbol{q}) = \left(\boldsymbol{p}_a + A(\theta_a)\boldsymbol{s}_a^{ab}\right) - \left(\boldsymbol{p}_b + A(\theta_b)\boldsymbol{s}_b^{ab}\right), \qquad \boldsymbol{p}_i = [x_i, y_i]^T. \tag{3.12}$$

Forward kinematics evaluates all five pairs of pin positions from the 15 body coordinates. A physically closed configuration is one for which every gap is zero. For a pin against ground, the ground attachment is a fixed world point and its term replaces the second body point. Chapter 5 assembles these gaps into $\Phi(\boldsymbol{q}) = \boldsymbol{0}$; the equations here define the geometric map, not a new coordinate choice for the dynamics.

The same loop can also be parameterized geometrically by two link orientations. This reduced parameterization is useful for position analysis and inverse-kinematics intuition; it does not replace the Cartesian coordinates in the constrained dynamics. Use a loop-local frame with $P_{34} = (0, 0)$ and $P_{45} = (l_4, 0)$, and define $\boldsymbol{e}(\alpha) = [\cos\alpha, \sin\alpha]^T$. Choose α_1 along link 1 from P_{51} to P_{12} and α_5 along link 5 from P_{45} to P_{51}. The first two moving pins follow directly:

$$P_{51} = P_{45} + l_5\boldsymbol{e}(\alpha_5), \qquad P_{12} = P_{51} + l_1\boldsymbol{e}(\alpha_1). \tag{3.13}$$

Define $\boldsymbol{n}(\alpha) = [-\sin\alpha, \cos\alpha]^T = d\boldsymbol{e}/d\alpha$. The target pin velocity and its Jacobian with respect to the independent angle rates are

$$\dot{P}_{12} = J_{12}\begin{bmatrix}\dot{\alpha}_1 \\ \dot{\alpha}_5\end{bmatrix}, \qquad J_{12} = \begin{bmatrix} l_1\boldsymbol{n}(\alpha_1) & l_5\boldsymbol{n}(\alpha_5)\end{bmatrix}, \qquad \det J_{12} = l_1 l_5 \sin(\alpha_5 - \alpha_1). \tag{3.14}$$

The target map loses local invertibility when links 1 and 5 are collinear. The remaining pin P_{23} must be at distance l_2 from P_{12} and distance l_3 from grounded pin P_{34}. Set $d = \|P_{34} - P_{12}\|$, $\boldsymbol{u} = (P_{34} - P_{12})/d$, and $\boldsymbol{n} = [-u_y, u_x]^T$. Circle intersection gives

$$a = \frac{l_2^2 - l_3^2 + d^2}{2d}, \qquad h = \sqrt{l_2^2 - a^2}, \qquad P_{23} = P_{12} + a\boldsymbol{u} + \sigma_{23}h\boldsymbol{n}, \quad \sigma_{23} \in \{-1, +1\}. \tag{3.15}$$

The formula gives $\|P_{23} - P_{12}\|^2 = a^2 + h^2 = l_2^2$ and $\|P_{34} - P_{23}\|^2 = (d-a)^2 + h^2 = l_3^2$. For $d > 0$, a real assembly requires $|l_2 - l_3| \le d \le l_2 + l_3$. At equality, the two circle branches meet and the middle links are collinear. If $d = 0$ and $l_2 = l_3$, every point on the common circle is an intersection, a degenerate singular case not covered by the formula. Each branch sign otherwise selects one of the two possible assemblies of links 2 and 3.

Differentiate the two fixed-length conditions to obtain the middle pin velocity. Let $\boldsymbol{d}_2 = P_{23} - P_{12}$ and $\boldsymbol{d}_3 = P_{23} - P_{34}$. Since P_{34} is fixed,

$$\begin{bmatrix} \boldsymbol{d}_2^T \\ \boldsymbol{d}_3^T \end{bmatrix} \dot{P}_{23} = \begin{bmatrix} \boldsymbol{d}_2^T \dot{P}_{12} \\ 0 \end{bmatrix}. \tag{3.16}$$

This solve determines $\dot{P}_{23}$ when the middle links are not collinear. Its matrix determinant is the signed-area factor $\boldsymbol{d}_2 \times \boldsymbol{d}_3$, which vanishes when the two middle links straighten or fold.

To convert the pins back to the 15 Cartesian body coordinates used by the dynamics, each link center is the midpoint of its ordered endpoints and its body angle is the direction from its left pin to its right pin:

$$\begin{aligned}
\boldsymbol{p}_1 &= \tfrac{1}{2}(P_{51} + P_{12}), \quad \theta_1 = \operatorname{atan2}((P_{12} - P_{51})_y, (P_{12} - P_{51})_x), \\
\boldsymbol{p}_2 &= \tfrac{1}{2}(P_{12} + P_{23}), \quad \theta_2 = \operatorname{atan2}((P_{23} - P_{12})_y, (P_{23} - P_{12})_x), \\
\boldsymbol{p}_3 &= \tfrac{1}{2}(P_{23} + P_{34}), \quad \theta_3 = \operatorname{atan2}((P_{34} - P_{23})_y, (P_{34} - P_{23})_x), \\
\boldsymbol{p}_4 &= \tfrac{1}{2}(P_{34} + P_{45}), \quad \theta_4 = 0, \\
\boldsymbol{p}_5 &= \tfrac{1}{2}(P_{45} + P_{51}), \quad \theta_5 = \operatorname{atan2}((P_{51} - P_{45})_y, (P_{51} - P_{45})_x).
\end{aligned} \tag{3.17}$$

Thus $(\alpha_1, \alpha_5, \sigma_{23})$ parameterizes a closed pose, while the constructed 15-vector still obeys the five pin closures and fixed-link pose. The geometric assembly fails at the circle-intersection limits, where $h = 0$ and the two branches merge.

For inverse position analysis, prescribe a target location $T = P_{12}$ for the pin shared by links 1 and 2. The other two unknown pins are found independently by intersecting two pairs of circles:

- P_{23} is the intersection of radius-l_2 circle centered at T and radius-l_3 circle centered at P_{34}.
- P_{51} is the intersection of radius-l_1 circle centered at T and radius-l_5 circle centered at P_{45}.

For either pair of centers C_a, C_b and radii r_a, r_b, let $d = \|C_b - C_a\|$, $\boldsymbol{u} = (C_b - C_a)/d$, $\boldsymbol{n} = [-u_y, u_x]^T$, and

$$a = \frac{r_a^2 - r_b^2 + d^2}{2d}, \qquad h = \sqrt{r_a^2 - a^2}, \qquad P = C_a + a\boldsymbol{u} + \sigma h \boldsymbol{n}, \quad \sigma \in \{-1, +1\}. \tag{3.18}$$

Apply this once for P_{23} and once for P_{51}. Both center distances must be nonzero for the formula. The first pair is feasible only if $|l_2 - l_3| \le \|P_{34} - T\| \le l_2 + l_3$; the second only if $|l_1 - l_5| \le \|P_{45} - T\| \le l_1 + l_5$. At a zero center distance with equal radii, the circle has infinitely many intersections and the pose is degenerate. Otherwise, the independent signs can produce up to four assemblies. For each feasible pair of intersections, recover all body poses using (3.17). This is inverse position kinematics for the specified target pin; a different task point or prescribed orientation gives a different target condition and may have a different number of solutions.

3.7 Summary

Forward kinematics maps body poses or independent link coordinates to world points. Inverse kinematics recovers joint angles or compatible loop assemblies from target positions and must account for feasibility and branch multiplicity. The double pendulum provides the simple two-link case; the five-link construction makes the closed-loop assembly branches explicit while retaining Cartesian body coordinates for dynamics. The next chapter differentiates the five pin gaps and derives the loop Jacobian and its acceleration-level terms.

Chapter 4

Deriving Dynamics and Preparing an ODE

4.1 State of the Art

The derivation in this chapter follows standard analytical mechanics and uses symbolic computation as an algebraic aid, not as a replacement for the mechanical model. Goldstein et al. provide a foundational analytical-mechanics treatment; SymPy documents the symbolic differentiation and expression manipulation used by the executable derivation [18, 19]. The numerical example uses an adaptive initial-value solver whose integration method, error tolerances, and output sampling are distinct settings [15]. This focused review covers the foundations and software behavior directly used here; it does not survey recent symbolic multibody generation systems. Bibliographic and software scope checked 2026-10-07.

4.2 Coordinates and modeling assumptions

Consider two point masses m_1 and m_2 connected by massless rigid rods of lengths l_1 and l_2. The first pivot is fixed. Rod and pivot friction are neglected, and applied torques are initially zero. This is an *open-chain* teaching example: the second mass is not connected back to ground, so there is no loop-closure constraint or Lagrange multiplier.

Both angles are absolute, positive clockwise from the downward vertical; world y is positive upward. The generalized coordinate and velocity vectors are

$$\boldsymbol{q} = [\theta_1, \theta_2]^T, \tag{4.1}$$

$$\dot{\boldsymbol{q}} = [\dot{\theta}_1, \dot{\theta}_2]^T, \qquad \ddot{\boldsymbol{q}} = [\ddot{\theta}_1, \ddot{\theta}_2]^T. \tag{4.2}$$

The lengths, masses, and gravity are kept symbolic during derivation and assigned numerical SI values only for integration.

Figure 4.1 uses the same convention as the equations and the validation notebook.

4.3 Deriving the equations with Lagrange's method

For a system with generalized coordinates $\boldsymbol{q}$, define the Lagrangian $L = T - V$, where T is kinetic energy and V is potential energy. For each coordinate q_i, the Euler–Lagrange

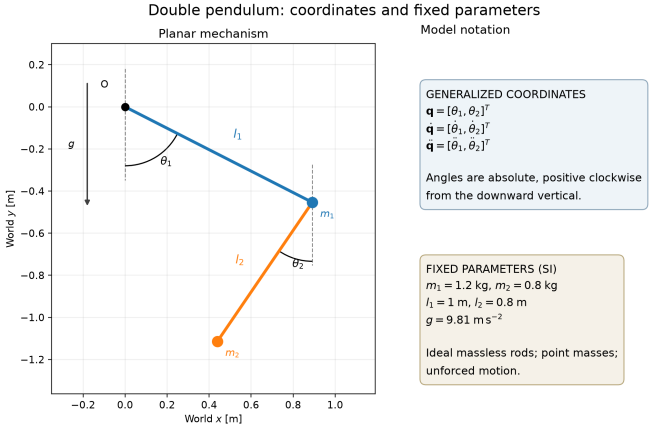


Figure 4.1: Double-pendulum geometry at the initial configuration. Angles are absolute and positive clockwise from the downward vertical; y is positive upward.

equation with applied generalized force Q_i is

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i. \tag{4.3}$$

The derivation proceeds by expressing positions and velocities in terms of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$, forming T and V, and applying (4.3). Collect the result as

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{Q}_{\text{ext}}, \tag{4.4}$$

where M multiplies the accelerations and $\boldsymbol{h}$ contains velocity-coupling and potential-energy terms. Here $\boldsymbol{Q}_{\text{ext}}$ denotes applied nonconservative generalized forces, such as actuator torques; gravity is included through V and therefore appears in $\boldsymbol{h}$.

4.3.1 Positions, velocities, and energy

The mass positions follow directly from the angle convention:

$$\begin{aligned} x_1 &= l_1 \sin\theta_1, & y_1 &= -l_1 \cos\theta_1, \\ x_2 &= x_1 + l_2 \sin\theta_2, & y_2 &= y_1 - l_2 \cos\theta_2. \end{aligned} \tag{4.5}$$

Differentiating with respect to time gives the Cartesian velocities:

$$\begin{aligned} \dot{x}_1 &= l_1 \cos\theta_1 \dot{\theta}_1, & \dot{y}_1 &= l_1 \sin\theta_1 \dot{\theta}_1, \\ \dot{x}_2 &= \dot{x}_1 + l_2 \cos\theta_2 \dot{\theta}_2, & \dot{y}_2 &= \dot{y}_1 + l_2 \sin\theta_2 \dot{\theta}_2. \end{aligned} \tag{4.6}$$

For clarity, define $\boldsymbol{v}_i = [\dot{x}_i, \dot{y}_i]^T$ and $\delta = \theta_1 - \theta_2$. Squaring the first velocity gives

$$\dot{x}_1^2 + \dot{y}_1^2 = l_1^2(\cos^2\theta_1 + \sin^2\theta_1)\dot{\theta}_1^2 = l_1^2\dot{\theta}_1^2. \tag{4.7}$$

For the second mass, use $\boldsymbol{v}_2 = \boldsymbol{v}_1 + l_2[\cos\theta_2, \sin\theta_2]^T\dot{\theta}_2$. Expanding the dot product separates the two individual speeds from their coupling:

$$\begin{aligned} \boldsymbol{v}_2 \cdot \boldsymbol{v}_2 &= \boldsymbol{v}_1 \cdot \boldsymbol{v}_1 + l_2^2\dot{\theta}_2^2 + 2l_2\dot{\theta}_2\left(\dot{x}_1\cos\theta_2 + \dot{y}_1\sin\theta_2\right) \\ &= l_1^2\dot{\theta}_1^2 + l_2^2\dot{\theta}_2^2 + 2l_1l_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2), \end{aligned} \tag{4.8}$$

The last equality follows from $\cos\theta_1\cos\theta_2+\sin\theta_1\sin\theta_2=\cos(\theta_1-\theta_2)$. Substituting (4.7)–(4.8) into $T=\frac{1}{2}m_1\boldsymbol{v}_1\cdot\boldsymbol{v}_1+\frac{1}{2}m_2\boldsymbol{v}_2\cdot\boldsymbol{v}_2$ gives

$$T=\frac{1}{2}(m_1+m_2)l_1^2\dot\theta_1^2+\frac{1}{2}m_2l_2^2\dot\theta_2^2+m_2l_1l_2\dot\theta_1\dot\theta_2\cos(\theta_1-\theta_2), \tag{4.9}$$

$$V=-(m_1+m_2)gl_1\cos\theta_1-m_2gl_2\cos\theta_2, \qquad L=T-V. \tag{4.10}$$

4.3.2 Euler–Lagrange residual and equation extraction

For symbolic derivation, define the left-hand-side residual vector

$$\boldsymbol{E}=\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\boldsymbol{q}}}\right)-\frac{\partial L}{\partial\boldsymbol{q}}, \qquad \boldsymbol{E}=\boldsymbol{Q}_{\text{ext}}. \tag{4.11}$$

To see what is being differentiated, first compute the generalized momenta and the coordinate derivatives of L:

$$\begin{aligned}
\frac{\partial L}{\partial\dot\theta_1}&=(m_1+m_2)l_1^2\dot\theta_1+m_2l_1l_2\cos\delta\,\dot\theta_2,\\
\frac{\partial L}{\partial\dot\theta_2}&=m_2l_2^2\dot\theta_2+m_2l_1l_2\cos\delta\,\dot\theta_1,\\
\frac{\partial L}{\partial\theta_1}&=-m_2l_1l_2\dot\theta_1\dot\theta_2\sin\delta-(m_1+m_2)gl_1\sin\theta_1,\\
\frac{\partial L}{\partial\theta_2}&=m_2l_1l_2\dot\theta_1\dot\theta_2\sin\delta-m_2gl_2\sin\theta_2.
\end{aligned} \tag{4.12}$$

Because $\dot\delta=\dot\theta_1-\dot\theta_2$, apply the product rule to each momentum:

$$\begin{aligned}
\frac{d}{dt}\frac{\partial L}{\partial\dot\theta_1}&=(m_1+m_2)l_1^2\ddot\theta_1+m_2l_1l_2\left[\ddot\theta_2\cos\delta-\dot\theta_2\sin\delta(\dot\theta_1-\dot\theta_2)\right],\\
\frac{d}{dt}\frac{\partial L}{\partial\dot\theta_2}&=m_2l_2^2\ddot\theta_2+m_2l_1l_2\left[\ddot\theta_1\cos\delta-\dot\theta_1\sin\delta(\dot\theta_1-\dot\theta_2)\right].
\end{aligned} \tag{4.13}$$

Now subtract the coordinate derivatives in (4.12). In the first residual the mixed terms $-\dot\theta_1\dot\theta_2\sin\delta$ and $+\dot\theta_1\dot\theta_2\sin\delta$ cancel, leaving $+\dot\theta_2^2\sin\delta$; in the second they cancel, leaving $-\dot\theta_1^2\sin\delta$. Thus the two scalar equations are

$$\begin{aligned}
E_1&=(m_1+m_2)l_1^2\ddot\theta_1+m_2l_1l_2\cos\delta\,\ddot\theta_2+m_2l_1l_2\sin\delta\,\dot\theta_2^2+(m_1+m_2)gl_1\sin\theta_1=\tau_1,\\
E_2&=m_2l_1l_2\cos\delta\,\ddot\theta_1+m_2l_2^2\ddot\theta_2-m_2l_1l_2\sin\delta\,\dot\theta_1^2+m_2gl_2\sin\theta_2=\tau_2.
\end{aligned} \tag{4.14}$$

The time derivatives are evaluated by the chain rule, treating $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, and $\ddot{\boldsymbol{q}}$ as symbolic variables. Because $\boldsymbol{E}$ is affine in $\ddot{\boldsymbol{q}}$, extract

$$M(\boldsymbol{q})=\frac{\partial\boldsymbol{E}}{\partial\ddot{\boldsymbol{q}}}, \qquad \boldsymbol{h}=\boldsymbol{E}-M\ddot{\boldsymbol{q}}. \tag{4.15}$$

Then verify symbolically that $\boldsymbol{E}-(M\ddot{\boldsymbol{q}}+\boldsymbol{h})=\boldsymbol{0}$. In the notebook, SymPy performs these differentiations and simplifications while retaining m_1,m_2,l_1,l_2,g as symbols. This makes sign or algebra errors easier to localize before parameter values are substituted.

Let $\delta=\theta_1-\theta_2$. The resulting equations are

$$\begin{bmatrix}(m_1+m_2)l_1^2 & m_2l_1l_2\cos\delta\\ m_2l_1l_2\cos\delta & m_2l_2^2\end{bmatrix}\begin{bmatrix}\ddot\theta_1\\ \ddot\theta_2\end{bmatrix}+\begin{bmatrix}m_2l_1l_2\sin\delta\,\dot\theta_2^2+(m_1+m_2)gl_1\sin\theta_1\\ -m_2l_1l_2\sin\delta\,\dot\theta_1^2+m_2gl_2\sin\theta_2\end{bmatrix}=\begin{bmatrix}\tau_1\\ \tau_2\end{bmatrix}. \tag{4.16}$$

The leading matrix is the configuration-dependent mass matrix; the remaining vector contains velocity-coupling and gravity terms. The unforced case used below sets $\tau_1 = \tau_2 = 0$. The symbolic notebook verifies that its extracted matrix and remainder agree with (4.16).

The torque components are positive in the directions of increasing θ_1 and θ_2; because these angles increase clockwise from downward vertical, this fixes the sign of any later actuator model. In the present example both are zero. The determinant of the mass matrix is

$$\det M = m_2 l_1^2 l_2^2 \left[(m_1 + m_2) - m_2 \cos^2 \delta\right] = m_2 l_1^2 l_2^2 \left(m_1 + m_2 \sin^2 \delta\right) > 0, \tag{4.17}$$

for positive masses and nonzero rod lengths. Therefore this model's acceleration solve is nonsingular at every configuration; this does not imply that every output-velocity inverse problem is nonsingular, as the tip Jacobian can still lose rank at collinear poses.

4.4 Preparing the first-order ODE

At each state, solve the linear system

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} = \boldsymbol{Q}_{\text{ext}} - \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{4.18}$$

for $\ddot{\boldsymbol{q}}$; do not explicitly form M^{-1}. Introduce the first-order state

$$\boldsymbol{y} = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]^T, \qquad \dot{\boldsymbol{y}} = [\dot{\theta}_1, \dot{\theta}_2, \ddot{\theta}_1, \ddot{\theta}_2]^T. \tag{4.19}$$

The ODE right-hand side therefore has four entries: the two angular velocities followed by the two accelerations returned by (4.18).

The evaluation order at one solver call is:

1. Read $\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2$ from the current state $\boldsymbol{y}$.
2. Evaluate $M(\boldsymbol{q})$ and $\boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ using the current angles and angular velocities.
3. Solve $M\ddot{\boldsymbol{q}} = \boldsymbol{Q}_{\text{ext}} - \boldsymbol{h}$ as a linear system for the two angular accelerations. Do not form M^{-1} explicitly.
4. Return $[\dot{\theta}_1, \dot{\theta}_2, \ddot{\theta}_1, \ddot{\theta}_2]^T$ as $\dot{\boldsymbol{y}}$.

This separation matters: the state contains positions and velocities, while the accelerations are algebraic outputs of the instantaneous dynamics solve.

4.4.1 Language-neutral pseudocode

1. Define symbolic coordinates, velocities, accelerations, physical parameters, and the angle convention.
2. Derive mass positions and velocities; construct T, V, and $L = T - V$.
3. Form the Euler–Lagrange residual $\boldsymbol{E}$ and extract M and $\boldsymbol{h}$.
4. Simplify and assert the symbolic decomposition residual is zero; compare the result with the displayed equations.
5. Substitute the selected SI parameter values and create numerical evaluators for M and $\boldsymbol{h}$.
6. Given $(t, \boldsymbol{y})$, split $\boldsymbol{y}$ into $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$, solve for $\ddot{\boldsymbol{q}}$, and return $\dot{\boldsymbol{y}}$.
7. Integrate from the specified initial state; check solver status; compute energy error; and make the requested plots.

4.5 Numerical example and energy check

The validation notebook uses $m_1 = 1.2\,\mathrm{kg}$, $m_2 = 0.8\,\mathrm{kg}$, $l_1 = 1.0\,\mathrm{m}$, $l_2 = 0.8\,\mathrm{m}$, and $g = 9.81\,\mathrm{m\,s^{-2}}$. The initial state is $[1.1, -0.6, 0, 0]^T$, with angles in radians and zero initial angular velocities. It integrates the unforced system on $0 \le t \le 10\,\mathrm{s}$ using adaptive RK45, relative tolerance 10^{-9}, absolute tolerance 10^{-11}, maximum step 0.01 s, and 2,001 requested output times. Those times specify where the solver stores or reports values; they do not force its internal adaptive steps to occur at those times. These are example settings, not universal defaults.

Since no torque or dissipation is applied, $E = T + V$ is conserved in the continuous model. At each saved state, evaluate

$$E(t) = T(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) + V(\boldsymbol{q}(t)), \qquad \varepsilon_E(t) = \frac{E(t) - E(0)}{\max(1\,\mathrm{J}, |E(0)|)}. \tag{4.20}$$

The validated numerical run has maximum absolute scaled energy error $\max_t |\varepsilon_E(t)| = 9.373 \times 10^{-9}$. This checks numerical consistency for this run; it is not proof that the idealized model represents a physical apparatus. Small energy error also does not guarantee small long-time phase or trajectory error in a nonlinear system. Figure 4.2 plots both angles and the signed scaled energy error. As a tolerance and step-size sensitivity check, the notebook repeats the same initial-value problem with rtol $= 10^{-10}$, atol $= 10^{-12}$, and maximum internal step 0.005 s. This run uses 13,136 right-hand-side evaluations, reduces the maximum scaled energy error to 4.494×10^{-10}, and has maximum sampled state difference 4.736×10^{-8} from the baseline. This single refinement supports numerical sensitivity assessment for this run; it is not a convergence proof or a long-time phase-error bound.

4.6 From the open-chain example to the five-link closed chain

The double pendulum above has two minimal coordinates: every (θ_1, θ_2) gives a geometrically possible pose. The five-link mechanism is different. Chapter 2 defined its perimeter loop and grounded link; Chapter 3 introduced Cartesian body coordinates and attachment-point kinematics. Here those ingredients are combined from pin geometry through the first-order simulation equation. The derivation follows the canonical notebook `constrained_five_link_dynamics.ipynb`; Chapters 5–9 revisit constraints, initialization, and drift control in detail.

4.6.1 Coordinates, pin locations, and position closure

For body i, let $\boldsymbol{p}_i = [x_i, y_i]^T$ be its center-of-mass position and let θ_i be its counterclockwise orientation from world x. Stack the five poses in body order:

$$\boldsymbol{q} = [x_1, y_1, \theta_1, x_2, y_2, \theta_2, \ldots, x_5, y_5, \theta_5]^T \in \mathbb{R}^{15}. \tag{4.21}$$

The symbolic derivation therefore has 15 generalized coordinates and retains the five link lengths $L_1, \ldots, L_5$ as symbols. The notebook's fixed reference pose for link 4 is initialized from its configured numerical pin geometry, then rationalized as a constant in the symbolic fixed-pose residual; the link lengths themselves remain symbolic. Numerical link lengths are substituted for the later cross-check and simulation. The rotation matrix and quarter-turn matrix are

$$A(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \qquad S = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \tag{4.22}$$

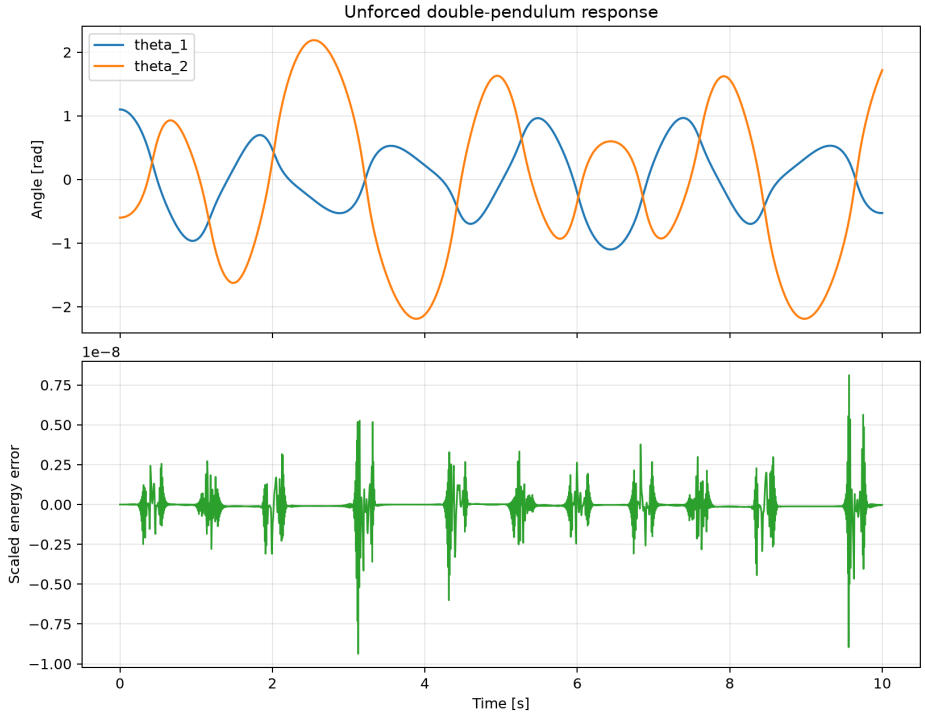


Figure 4.2: Angles and signed scaled energy error for the 10-second unforced double-pendulum run. The figure is generated by the validation notebook using the parameters and solver settings stated in the text.

Every link's center of mass is at the midpoint of its pin-to-pin segment. Thus the two local pin offsets of link i are $-\frac{1}{2}L_i[1,0]^T$ and $+\frac{1}{2}L_i[1,0]^T$. In cyclic order the pins are $P_{12}, P_{23}, P_{34}, P_{45}, P_{51}$. For example,

$$\begin{aligned} \boldsymbol{r}_{12}^{(1)} &= \boldsymbol{p}_1 + A(\theta_1)\tfrac{L_1}{2}\boldsymbol{e}_x, \quad \boldsymbol{r}_{12}^{(2)} = \boldsymbol{p}_2 - A(\theta_2)\tfrac{L_2}{2}\boldsymbol{e}_x, \\ \boldsymbol{r}_{51}^{(5)} &= \boldsymbol{p}_5 + A(\theta_5)\tfrac{L_5}{2}\boldsymbol{e}_x, \quad \boldsymbol{r}_{51}^{(1)} = \boldsymbol{p}_1 - A(\theta_1)\tfrac{L_1}{2}\boldsymbol{e}_x, \end{aligned} \qquad \boldsymbol{e}_x = [1,0]^T. \tag{4.23}$$

The remaining three pairs follow the same right-end-of-first-link, left-end-of-next-link rule. For every pin, closure requires its two attachment points to coincide. Define each two-component gap as first-listed attachment minus second-listed attachment and stack the five gaps, followed by the grounded-pose residual:

$$\boldsymbol{\Phi}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{r}_{12}^{(1)} - \boldsymbol{r}_{12}^{(2)} \\ \boldsymbol{r}_{23}^{(2)} - \boldsymbol{r}_{23}^{(3)} \\ \boldsymbol{r}_{34}^{(3)} - \boldsymbol{r}_{34}^{(4)} \\ \boldsymbol{r}_{45}^{(4)} - \boldsymbol{r}_{45}^{(5)} \\ \boldsymbol{r}_{51}^{(5)} - \boldsymbol{r}_{51}^{(1)} \\ \boldsymbol{p}_4 - \boldsymbol{p}_4^0 \\ \theta_4 - \theta_4^0 \end{bmatrix} = \boldsymbol{0}. \tag{4.24}$$

There are ten pin-gap components and three fixed-pose components. The notebook sets $\theta_4^0 = 0$ so link 4 is horizontal; $\boldsymbol{p}_4^0$ is its chosen reference center. This gives 13 written scalar equations, not by itself a proof that all 13 are independent. At the notebook's modeled pose, the Jacobian rank is 13 and the local mobility is $15 - 13 = 2$.

Expanding the five point differences gives the notebook's complete 13-entry symbolic residual. Here $c_i = \cos\theta_i$ and $s_i = \sin\theta_i$:

$$\boldsymbol{\Phi}(\boldsymbol{q}) = \begin{bmatrix} x_1 - x_2 + \frac{1}{2}L_1c_1 + \frac{1}{2}L_2c_2 \\ y_1 - y_2 + \frac{1}{2}L_1s_1 + \frac{1}{2}L_2s_2 \\ x_2 - x_3 + \frac{1}{2}L_2c_2 + \frac{1}{2}L_3c_3 \\ y_2 - y_3 + \frac{1}{2}L_2s_2 + \frac{1}{2}L_3s_3 \\ x_3 - x_4 + \frac{1}{2}L_3c_3 + \frac{1}{2}L_4c_4 \\ y_3 - y_4 + \frac{1}{2}L_3s_3 + \frac{1}{2}L_4s_4 \\ x_4 - x_5 + \frac{1}{2}L_4c_4 + \frac{1}{2}L_5c_5 \\ y_4 - y_5 + \frac{1}{2}L_4s_4 + \frac{1}{2}L_5s_5 \\ x_5 - x_1 + \frac{1}{2}L_5c_5 + \frac{1}{2}L_1c_1 \\ y_5 - y_1 + \frac{1}{2}L_5s_5 + \frac{1}{2}L_1s_1 \\ x_4 - x_4^0 \\ y_4 - y_4^0 \\ \theta_4 - \theta_4^0 \end{bmatrix} = \mathbf{0}. \tag{4.25}$$

In the configured notebook example, $(x_4^0, y_4^0, \theta_4^0) = (0, -0.4\,\mathrm{m}, 0)$, so the last three entries specialize to x_4, $y_4 + 0.4\,\mathrm{m}$, and θ_4. The symbolic link lengths remain $L_1, \ldots, L_5$; those example reference coordinates are not link-length substitutions.

The full 13×15 Jacobian from the notebook can be written sparsely in 3-column body blocks. Define the endpoint Jacobians for a right and left pin of body i by

$$B_i^+ = \begin{bmatrix} 1 & 0 & -\frac{1}{2}L_is_i \\ 0 & 1 & \frac{1}{2}L_ic_i \end{bmatrix}, \qquad B_i^- = \begin{bmatrix} 1 & 0 & \frac{1}{2}L_is_i \\ 0 & 1 & -\frac{1}{2}L_ic_i \end{bmatrix}. \tag{4.26}$$

The first ten rows are the ordered pin residual Jacobians, and the last three rows ground link 4:

$$J = \begin{bmatrix} B_1^+ & -B_2^- & 0 & 0 & 0 \\ 0 & B_2^+ & -B_3^- & 0 & 0 \\ 0 & 0 & B_3^+ & -B_4^- & 0 \\ 0 & 0 & 0 & B_4^+ & -B_5^- \\ -B_1^- & 0 & 0 & 0 & B_5^+ \\ 0 & 0 & 0 & I_3 & 0 \end{bmatrix} \in \mathbb{R}^{13\times 15}. \tag{4.27}$$

Each displayed pin row is a two-row block; thus the five pin blocks contribute ten rows and the grounding block contributes three. This block form is the same sparse matrix as $\partial\boldsymbol{\Phi}/\partial\boldsymbol{q}$, but exposes which bodies each pin couples.

4.6.2 Differentiate closure to obtain velocity and acceleration conditions

For a body-fixed point $\boldsymbol{s}_i$ on body i, its world position is $\boldsymbol{r}_i = \boldsymbol{p}_i + A(\theta_i)\boldsymbol{s}_i$. Differentiating A gives $\partial A/\partial\theta = SA$, so

$$\dot{\boldsymbol{r}}_i = \dot{\boldsymbol{p}}_i + \dot{\theta}_i SA(\theta_i)\boldsymbol{s}_i, \qquad \ddot{\boldsymbol{r}}_i = \ddot{\boldsymbol{p}}_i + \ddot{\theta}_i SA\boldsymbol{s}_i - \dot{\theta}_i^2 A\boldsymbol{s}_i. \tag{4.28}$$

The last term is centripetal acceleration of the rotating offset. For one pin residual $\boldsymbol{r}_a - \boldsymbol{r}_b$, its Jacobian has the body blocks

$$J_a = \begin{bmatrix} I_2 & SA(\theta_a)\boldsymbol{s}_a \end{bmatrix}, \qquad J_b = -\begin{bmatrix} I_2 & SA(\theta_b)\boldsymbol{s}_b \end{bmatrix}, \tag{4.29}$$

with zeros in all other body columns. The three fixed-pose rows select (x_4, y_4, θ_4), so their Jacobian block is I_3. Stacking all rows gives

$$J(\boldsymbol{q}) = \frac{\partial\boldsymbol{\Phi}}{\partial\boldsymbol{q}} \in \mathbb{R}^{13\times 15}, \qquad \dot{\boldsymbol{\Phi}} = J\dot{\boldsymbol{q}} = \mathbf{0}. \tag{4.30}$$

Differentiate once more:

$$J\ddot{\boldsymbol{q}} + \dot{J}\dot{\boldsymbol{q}} = \mathbf{0}. \tag{4.31}$$

Because fixed-pose rows are linear, their contribution to $\dot{J}\dot{\boldsymbol{q}}$ is zero. For a pin gap, only the rotating attachment offsets contribute curvature:

$$\boldsymbol{\gamma}_{ab} = (\dot{J}\dot{\boldsymbol{q}})_{ab} = -\dot{\theta}_a^2 A(\theta_a)\boldsymbol{s}_a + \dot{\theta}_b^2 A(\theta_b)\boldsymbol{s}_b. \tag{4.32}$$

The signs match the residual convention: the first attachment enters positively and the second negatively. This acceleration bias is determined by geometry and velocity; it is not a gravity or applied-force term.

4.6.3 Energies and ideal multiplier equations

With each origin at a body center of mass, the kinetic and gravitational potential energies are

$$T = \frac{1}{2}\sum_{i=1}^{5}\left[m_i(\dot{x}_i^2 + \dot{y}_i^2) + I_i\dot{\theta}_i^2\right] = \frac{1}{2}\dot{\boldsymbol{q}}^T M\dot{\boldsymbol{q}}, \qquad V = \sum_{i=1}^{5} m_i g y_i, \tag{4.33}$$

where

$$M = \operatorname{diag}(m_1, m_1, I_1, \ldots, m_5, m_5, I_5), \qquad \boldsymbol{Q}_g = -\frac{\partial V}{\partial \boldsymbol{q}}. \tag{4.34}$$

The symbolic notebook's diagonal mass matrix and expanded energies are therefore

$$M = \operatorname{diag}(m_1, m_1, I_1, m_2, m_2, I_2, m_3, m_3, I_3, m_4, m_4, I_4, m_5, m_5, I_5), \tag{4.35}$$

$$\begin{aligned} T &= \frac{1}{2}\sum_{i=1}^{5} I_i\dot{\theta}_i^2 + \frac{1}{2}m_1(\dot{x}_1^2 + \dot{y}_1^2) + \frac{1}{2}m_2(\dot{x}_2^2 + \dot{y}_2^2) \\ &\quad + \frac{1}{2}m_3(\dot{x}_3^2 + \dot{y}_3^2) + \frac{1}{2}m_4(\dot{x}_4^2 + \dot{y}_4^2) + \frac{1}{2}m_5(\dot{x}_5^2 + \dot{y}_5^2), \\ V &= g(m_1 y_1 + m_2 y_2 + m_3 y_3 + m_4 y_4 + m_5 y_5). \end{aligned} \tag{4.36}$$

Differentiating V with respect to the coordinate order in (4.21) gives the full generalized gravity vector

$$\boldsymbol{Q}_g = \begin{bmatrix} 0 \\ -gm_1 \\ 0 \\ 0 \\ -gm_2 \\ 0 \\ 0 \\ -gm_3 \\ 0 \\ 0 \\ -gm_4 \\ 0 \\ 0 \\ -gm_5 \\ 0 \end{bmatrix}. \tag{4.37}$$

Since these are center-of-mass coordinates and the bodies are rigid with fixed mass properties, M is constant and there is no velocity-dependent inertial bias. With no non-gravitational loads, $\boldsymbol{Q}_{\text{total}} = \boldsymbol{Q}_g$. Ideal pin and support reactions do virtual work only

in directions prohibited by the constraints; in redundant coordinates represent their generalized force by $J^T\boldsymbol{\lambda}$. Lagrange's equations become

$$M\ddot{\boldsymbol{q}} + J^T\boldsymbol{\lambda} = \boldsymbol{Q}_{\text{total}}, \qquad J\ddot{\boldsymbol{q}} = -\boldsymbol{\gamma}. \tag{4.38}$$

The first line is 15 force-balance equations. The second line supplies 13 acceleration-compatibility equations. Together they determine 15 accelerations and 13 multipliers:

$$\begin{bmatrix} M & J^T \\ J & 0_{13\times 13} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{Q}_{\text{total}} \\ -\boldsymbol{\gamma} \end{bmatrix}. \tag{4.39}$$

This is a 28×28 saddle-point solve. The lower rows are ideal acceleration constraints; no numerical stabilization gains have been included. Multiplier signs depend on the chosen gap order, but the physical reaction is unchanged when the residual and multiplier conventions are transformed consistently.

4.6.4 From the KKT solve to an ODE, and why closure drifts

If the state is $\boldsymbol{y} = [\boldsymbol{q}^T, \dot{\boldsymbol{q}}^T]^T \in \mathbb{R}^{30}$, solve (4.39) at each right-hand-side evaluation and return

$$\dot{\boldsymbol{y}} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{bmatrix}. \tag{4.40}$$

This is an ODE reformulation of the acceleration-level constrained problem, not a direct integration of all position-level algebraic equations by a DAE solver. In exact arithmetic, if initially $\boldsymbol{\Phi} = \boldsymbol{0}$ and $J\dot{\boldsymbol{q}} = \boldsymbol{0}$, the ideal acceleration equation keeps their time derivatives zero. A standard ODE stepper, however, controls estimated local state error; it does not project the accepted state back onto either constraint manifold. Truncation and roundoff errors can therefore leave nonzero $\boldsymbol{\Phi}$ or $J\dot{\boldsymbol{q}}$. The ideal acceleration equation supplies no restoring feedback for those position/velocity errors: an off-manifold state is not automatically drawn back to the correct closed configuration. The errors may persist or accumulate, and the resulting pose can become physically inconsistent even while the ODE solver reports success.

Thus "ODE divergence" here should not be read as an inevitable blow-up of every trajectory. The common failure is constraint drift or loss of accuracy, not necessarily unbounded coordinates. Small ODE tolerances alone do not guarantee small closure residuals, especially when the residual components have different scales or the Jacobian is poorly conditioned near a singular pose. A dedicated DAE method, position/velocity projection, or stabilization such as Baumgarte feedback can address drift in different ways; each has its own accuracy, conditioning, and stability tradeoffs. The notebook uses one initial projection and a Baumgarte-modified acceleration solve, then tracks pin closure, grounded translation and angle errors, and the corresponding velocity residuals separately throughout the trajectory. Chapters 8 and 9 treat consistent initialization and drift control separately.

For the notebook's baseline ten-second run, the maximum pin-position and pin-velocity residuals over saved states are $1.108\times10^{-9}\,\text{m}$ and $1.247\times10^{-7}\,\text{m}\,\text{s}^{-1}$. Grounded-link translation and angle errors are at most $1.333\times10^{-17}\,\text{m}$ and $3.031\times10^{-19}\,\text{rad}$; its translation-rate and angular-rate residuals are at most $1.092\times10^{-15}\,\text{m}\,\text{s}^{-1}$ and $1.648\times10^{-17}\,\text{s}^{-1}$. These are outcomes for the stated geometry, gains, solver, and tolerances, not universal thresholds or a proof of physical validity.

4.7 Summary and transition

The chapter has now followed two models through dynamics construction. The double pendulum shows the compact minimal-coordinate Lagrange route; the five-link model shows how Cartesian body coordinates, pin closure, differentiated constraints, and ideal reaction multipliers combine in a KKT acceleration solve. The latter can be supplied to an ODE integrator, but ordinary ODE error control alone does not preserve algebraic closure. Chapters 5–9 develop the constraint equations, consistent initialization, and drift-control choices in detail.

Chapter 5

Joint Constraints and Loop Closure

5.1 State of the Art

Closed-chain kinematics can be represented by holonomic equations requiring connected attachment points to coincide. Redundant Cartesian coordinates make these equations direct to assemble, while reduced-coordinate methods encode closure relations in the coordinates themselves. The choice depends on topology and singularity behavior [2]. This focused review is limited to the foundational formulation and rank interpretation used in the example; it does not compare recent loop-closure algorithms. Bibliographic scope checked 2026-10-07.

5.2 Revolute-pin constraint

A planar revolute pin joins one point fixed in body a to one point fixed in body b. Let $\boldsymbol{s}_a$ and $\boldsymbol{s}_b$ be the respective body-frame attachment coordinates. Using the point map from Chapter 3, the world positions are

$$\boldsymbol{r}_a = \boldsymbol{p}_a + A(\theta_a)\boldsymbol{s}_a, \qquad \boldsymbol{r}_b = \boldsymbol{p}_b + A(\theta_b)\boldsymbol{s}_b, \tag{5.1}$$

where $\boldsymbol{p}_i = [x_i, y_i]^T$. Coincidence of the points gives the two scalar holonomic constraints

$$\boldsymbol{\Phi}_{ab}(\boldsymbol{q}) = \boldsymbol{r}_a(\boldsymbol{q}) - \boldsymbol{r}_b(\boldsymbol{q}) = \boldsymbol{0}. \tag{5.2}$$

The sign convention here is first body minus second body. Reversing the residual sign is equivalent if the Jacobian and multiplier convention are changed consistently.

The Jacobian of this pin residual has nonzero blocks only for the two connected bodies:

$$\frac{\partial \boldsymbol{\Phi}_{ab}}{\partial \boldsymbol{q}_a} = \begin{bmatrix} I_2 & SA(\theta_a)\boldsymbol{s}_a \end{bmatrix}, \qquad \frac{\partial \boldsymbol{\Phi}_{ab}}{\partial \boldsymbol{q}_b} = -\begin{bmatrix} I_2 & SA(\theta_b)\boldsymbol{s}_b \end{bmatrix}. \tag{5.3}$$

These blocks follow by differentiating the attachment-point positions with respect to each body's (x, y, θ) coordinates.

5.3 Grounding and assembling the loop

Fixing a planar rigid body at reference pose $\boldsymbol{q}_k^*$ contributes three scalar constraints:

$$\boldsymbol{\Phi}_{\mathrm{fix},k}(\boldsymbol{q}) = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} - \boldsymbol{q}_k^* = \boldsymbol{0}. \tag{5.4}$$

For the five-link example, the pin graph is the perimeter cycle

$$(1,2),\ (2,3),\ (3,4),\ (4,5),\ (5,1), \tag{5.5}$$

and link 4 is grounded at its initial horizontal pose. Stack the five two-component pin residuals and the three fixed-pose residuals, in a documented row order, to form

$$\boldsymbol{\Phi}(\boldsymbol{q}) = [\boldsymbol{\Phi}_{12}^T, \boldsymbol{\Phi}_{23}^T, \boldsymbol{\Phi}_{34}^T, \boldsymbol{\Phi}_{45}^T, \boldsymbol{\Phi}_{51}^T, \boldsymbol{\Phi}_{\text{fix},4}^T]^T = \mathbf{0}. \tag{5.6}$$

This vector has 13 scalar entries: ten from pins and three from grounding. The mechanism has 15 Cartesian coordinates, but the number of constraint entries alone does not establish the number of independent constraints.

5.3.1 Assembly procedure used by the symbolic model

The canonical five-link notebook keeps the link lengths symbolic while constructing the residual and Jacobian. With every body frame at its center of mass and aligned with its link, the left and right attachment coordinates of link i are $-\frac{1}{2}L_i\boldsymbol{e}_x$ and $+\frac{1}{2}L_i\boldsymbol{e}_x$. The following procedure describes the complete assembly without depending on a programming language:

1. Order body coordinates as (x_i, y_i, θ_i) for $i = 1, \ldots, 5$; define the cyclic pin list $(1,2), (2,3), (3,4), (4,5), (5,1)$.
2. For each listed pin, map the right endpoint of the first link and the left endpoint of the second link into world coordinates using (5.1); append their two-component difference to $\boldsymbol{\Phi}$.
3. Append the three residuals fixing the center position and orientation of link 4 to its reference pose.
4. Differentiate the assembled residual symbolically with respect to all 15 coordinates to form $J = \partial\boldsymbol{\Phi}/\partial\boldsymbol{q}$. Keep $L_1, \ldots, L_5$ symbolic rather than substituting rounded geometric lengths.
5. Check that $\boldsymbol{\Phi}$ has 13 entries and J has dimensions 13×15. At a projected representative pose, evaluate $\operatorname{rank}(J)$ and confirm the expected local rank before using $15 - \operatorname{rank}(J)$ as the mobility.

The notebook also constructs the same residual and Jacobian in a numerical kinematics class. Evaluating both implementations at the same projected state checks endpoint ordering, residual signs, and the fixed-link rows independently of the symbolic differentiation. The demonstrated differences are below 3.4×10^{-16} for $\boldsymbol{\Phi}$ and 2.3×10^{-16} for J at that state; these are implementation checks for this pose, not general conditioning guarantees.

5.4 Constraint Jacobian, rank, and local mobility

Differentiate the residual vector with respect to generalized coordinates:

$$J(\boldsymbol{q}) = \frac{\partial\boldsymbol{\Phi}}{\partial\boldsymbol{q}}. \tag{5.7}$$

At a regular configuration, local mobility is the number of coordinates minus the Jacobian rank:

$$n_{\text{dof}} = n_q - \operatorname{rank} J(\boldsymbol{q}). \tag{5.8}$$

For the notebook's selected pose, $n_q = 15$ and $\operatorname{rank} J = 13$, giving two local degrees of freedom. This rank statement applies at the evaluated pose; singular configurations can change the rank and hence the local mobility. The SymPy derivation retains link lengths symbolically, while the notebook numerically evaluates rank at the chosen geometry.

5.5 Summary and transition

Each planar pin contributes two point-coincidence equations, and grounding one body contributes three pose equations. Stacking the five perimeter pins and the ground constraints defines $\boldsymbol{\Phi}(\boldsymbol{q}) = \mathbf{0}$; the Jacobian rank, not equation count alone, determines local mobility. The next chapter differentiates these position constraints to obtain velocity and acceleration compatibility.

Chapter 6

Constraint Velocities and Accelerations

6.1 State of the Art

Constraint kinematics follow by differentiating holonomic position equations. Evaluation of the acceleration bias and Jacobian derivatives is important for simulation and reaction-force calculations [2]. This focused review uses the standard analytical derivative hierarchy for the planar pin model; it does not compare automatic-differentiation or recursive implementations. Bibliographic scope checked 2026-10-07.

6.2 Position constraints imply velocity constraints

Begin with the assembled holonomic equations

$$\boldsymbol{\Phi}(\boldsymbol{q}) = \mathbf{0}. \tag{6.1}$$

Define the constraint Jacobian $J(\boldsymbol{q}) = \partial\boldsymbol{\Phi}/\partial\boldsymbol{q}$. Differentiating (6.1) with respect to time gives

$$\dot{\boldsymbol{\Phi}} = J(\boldsymbol{q})\dot{\boldsymbol{q}} = \mathbf{0}. \tag{6.2}$$

The admissible generalized velocities therefore lie in the null space of J. At a regular configuration, that null space has dimension $n_q - \operatorname{rank}(J)$, equal to the local mobility.

6.3 Acceleration constraints and curvature terms

Differentiate (6.2) again. Because J depends on configuration, its time derivative contributes:

$$\ddot{\boldsymbol{\Phi}} = J(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{J}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \mathbf{0}. \tag{6.3}$$

The vector $\dot{J}\dot{\boldsymbol{q}}$ is sometimes called the acceleration bias or curvature term. It is generally nonzero even though the position constraints are homogeneous and time-independent.

For a point fixed in a rotating planar body, write its body-frame offset as $\boldsymbol{s}$ and let $\boldsymbol{\rho} = A(\theta)\boldsymbol{s}$ be the rotated offset. Since $\dot{\boldsymbol{\rho}} = \dot{\theta}S\boldsymbol{\rho}$, its world acceleration is

$$\ddot{\boldsymbol{r}} = \ddot{\boldsymbol{p}} + \ddot{\theta}S\boldsymbol{\rho} - \dot{\theta}^2\boldsymbol{\rho}. \tag{6.4}$$

The last term is centripetal acceleration. For a pin residual $\boldsymbol{r}_a - \boldsymbol{r}_b$, take the signed difference of the two attached-point accelerations. The terms multiplying translational

and angular accelerations form $J\ddot{\boldsymbol{q}}$; the remaining squared-angular-speed terms form $\dot{J}\dot{\boldsymbol{q}}$. Fixed-pose constraints are linear in the coordinates and contribute no curvature term. If $\boldsymbol{\rho}_a = A(\theta_a)\boldsymbol{s}_a$ and $\boldsymbol{\rho}_b = A(\theta_b)\boldsymbol{s}_b$ are the world attachment offsets, the two-component pin bias is

$$\boldsymbol{\gamma}_{ab} = -\dot{\theta}_a^2\boldsymbol{\rho}_a + \dot{\theta}_b^2\boldsymbol{\rho}_b. \tag{6.5}$$

The signs follow from subtracting the centripetal terms $-\dot{\theta}_i^2\boldsymbol{\rho}_i$ in the two point accelerations according to the first-body-minus-second-body residual convention.

6.4 Five-link constraint derivative hierarchy

For the five-link model, the complete hierarchy is

$$\begin{aligned} \text{position:} \quad & \boldsymbol{\Phi}(\boldsymbol{q}) = \boldsymbol{0}, \\ \text{velocity:} \quad & J(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{0}, \\ \text{acceleration:} \quad & J(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{\gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{0}, \end{aligned} \tag{6.6}$$

where $\boldsymbol{\gamma} = \dot{J}\dot{\boldsymbol{q}}$. The symbolic notebook derives this term by differentiating the velocity-level equations; the numerical model independently computes the same centripetal contributions. Their agreement is checked at the projected initial state before integration. The notebook computes it in two independent ways. Symbolically, form $\dot{\boldsymbol{\Phi}} = J\dot{\boldsymbol{q}}$, differentiate with respect to $\boldsymbol{q}$ and multiply by $\dot{\boldsymbol{q}}$; this is the part of the total derivative that excludes $J\ddot{\boldsymbol{q}}$. Numerically, evaluate (6.5) for each pin and append zeros for the fixed-pose rows. Compare these vectors at the same consistent state before integration. At the notebook's projected initial state, the maximum component difference is 4.337×10^{-19}.

For the symbolic notebook's midpoint links, each pin residual joins the right endpoint of link i to the left endpoint of link j. Write $c_i = \cos\theta_i$, $s_i = \sin\theta_i$, and $\omega_i = \dot{\theta}_i$. The 13 ideal acceleration residuals, ordered as the two rows for each pin followed by the grounded pose, are

$$\begin{aligned}
r_{12,x} &= \ddot{x}_1 - \ddot{x}_2 - \frac{L_1}{2}(\ddot{\theta}_1 s_1 + \omega_1^2 c_1) - \frac{L_2}{2}(\ddot{\theta}_2 s_2 + \omega_2^2 c_2) = 0, \\
r_{12,y} &= \ddot{y}_1 - \ddot{y}_2 + \frac{L_1}{2}(\ddot{\theta}_1 c_1 - \omega_1^2 s_1) + \frac{L_2}{2}(\ddot{\theta}_2 c_2 - \omega_2^2 s_2) = 0, \\
r_{23,x} &= \ddot{x}_2 - \ddot{x}_3 - \frac{L_2}{2}(\ddot{\theta}_2 s_2 + \omega_2^2 c_2) - \frac{L_3}{2}(\ddot{\theta}_3 s_3 + \omega_3^2 c_3) = 0, \\
r_{23,y} &= \ddot{y}_2 - \ddot{y}_3 + \frac{L_2}{2}(\ddot{\theta}_2 c_2 - \omega_2^2 s_2) + \frac{L_3}{2}(\ddot{\theta}_3 c_3 - \omega_3^2 s_3) = 0, \\
r_{34,x} &= \ddot{x}_3 - \ddot{x}_4 - \frac{L_3}{2}(\ddot{\theta}_3 s_3 + \omega_3^2 c_3) - \frac{L_4}{2}(\ddot{\theta}_4 s_4 + \omega_4^2 c_4) = 0, \\
r_{34,y} &= \ddot{y}_3 - \ddot{y}_4 + \frac{L_3}{2}(\ddot{\theta}_3 c_3 - \omega_3^2 s_3) + \frac{L_4}{2}(\ddot{\theta}_4 c_4 - \omega_4^2 s_4) = 0, \\
r_{45,x} &= \ddot{x}_4 - \ddot{x}_5 - \frac{L_4}{2}(\ddot{\theta}_4 s_4 + \omega_4^2 c_4) - \frac{L_5}{2}(\ddot{\theta}_5 s_5 + \omega_5^2 c_5) = 0, \\
r_{45,y} &= \ddot{y}_4 - \ddot{y}_5 + \frac{L_4}{2}(\ddot{\theta}_4 c_4 - \omega_4^2 s_4) + \frac{L_5}{2}(\ddot{\theta}_5 c_5 - \omega_5^2 s_5) = 0, \\
r_{51,x} &= \ddot{x}_5 - \ddot{x}_1 - \frac{L_5}{2}(\ddot{\theta}_5 s_5 + \omega_5^2 c_5) - \frac{L_1}{2}(\ddot{\theta}_1 s_1 + \omega_1^2 c_1) = 0, \\
r_{51,y} &= \ddot{y}_5 - \ddot{y}_1 + \frac{L_5}{2}(\ddot{\theta}_5 c_5 - \omega_5^2 s_5) + \frac{L_1}{2}(\ddot{\theta}_1 c_1 - \omega_1^2 s_1) = 0, \\
r_{g,x} &= \ddot{x}_4 = 0, \qquad r_{g,y} = \ddot{y}_4 = 0, \qquad r_{g,\theta} = \ddot{\theta}_4 = 0.
\end{aligned} \tag{6.7}$$

Collecting these entries gives exactly $J\ddot{\boldsymbol{q}}+\dot{J}\dot{\boldsymbol{q}}=\mathbf{0}$. The terms proportional to $\ddot{\theta}_i$ come from tangential acceleration of the endpoint offsets; the ω_i^2 terms are centripetal contributions. In the notebook's stabilized solve, Baumgarte feedback is formed over the full 13-entry residual, including the three grounded-pose rows. Those final three correction terms are zero while link 4 remains exactly at its reference pose with zero velocity; they are not omitted from the formula.

Equation (6.3) supplies the kinematic equations needed to determine accelerations compatible with loop closure. In the next chapter, these equations are coupled to force balance, with Lagrange multipliers representing ideal joint reactions.

6.5 Summary

Differentiating holonomic constraints produces velocity compatibility and then acceleration compatibility. The acceleration equation contains both the unknown generalized accelerations and the configuration/velocity-dependent bias term. Omitting that term would fail to enforce the actual acceleration of rotating pin locations.

Chapter 7

Closed-Chain Dynamics and Reaction Forces

7.1 State of the Art

Constrained multibody formulations combine rigid-body force balance with algebraic joint constraints. Lagrange multipliers represent ideal reactions when redundant coordinates are retained [2]. The resulting saddle-point equations are related to differential-algebraic formulations, whose numerical treatment depends on consistency and solver choices [5]. This focused review covers the standard multiplier formulation and its DAE context; it does not benchmark current constrained-dynamics algorithms. Bibliographic scope checked 2026-10-07.

7.2 Force balance for Cartesian rigid bodies

For the five-link model, each planar body is described by its center-of-mass coordinates and orientation. With constant mass and planar inertia, the mass matrix is

$$M = \operatorname{diag}(m_1, m_1, I_1, \ldots, m_5, m_5, I_5). \tag{7.1}$$

In these center-of-mass coordinates, the kinetic energy is quadratic with constant M, so there is no velocity-dependent inertial bias in the unconstrained force-balance equations.

Let $\boldsymbol{Q}_{\text{ext}}$ denote applied generalized forces other than gravity, and let $V(\boldsymbol{q})$ be the gravitational potential. The gravitational generalized force is

$$\boldsymbol{Q}_g = -\frac{\partial V}{\partial \boldsymbol{q}}, \qquad \boldsymbol{Q}_{\text{total}} = \boldsymbol{Q}_{\text{ext}} + \boldsymbol{Q}_g. \tag{7.2}$$

For example, with y_i positive upward, $V = \sum_i m_i g y_i$ and each vertical force component is $-m_i g$.

Without constraints, force balance would be $M\ddot{\boldsymbol{q}} = \boldsymbol{Q}_{\text{total}}$. Joint reactions must be added so the accelerations also satisfy loop closure.

7.3 Lagrange multipliers as ideal reactions

Let $\boldsymbol{\Phi}(\boldsymbol{q}) = \boldsymbol{0}$ be the vector of holonomic pin and fixed-pose constraints, with Jacobian $J = \partial\boldsymbol{\Phi}/\partial\boldsymbol{q}$. Under the sign convention used in this book, the generalized ideal reaction

force is $J^T\boldsymbol{\lambda}$. The equations of motion are therefore

$$M\ddot{\boldsymbol{q}} + J^T\boldsymbol{\lambda} = \boldsymbol{Q}_{\text{total}}. \tag{7.3}$$

The multiplier vector contains two components for each planar pin and three components for the grounded pose. Its entries depend on the chosen residual signs and row ordering; the physical reactions are unchanged if conventions are transformed consistently.

The acceleration constraints from Chapter 6 are

$$J\ddot{\boldsymbol{q}} + \boldsymbol{\gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{0}, \qquad \boldsymbol{\gamma} = \dot{J}\dot{\boldsymbol{q}}. \tag{7.4}$$

Equations (7.3) and (7.4) determine both the body accelerations and the constraint reactions.

7.4 Acceleration-level KKT system

Stacking force balance and acceleration compatibility gives the saddle-point system

$$\begin{bmatrix} M & J^T \\ J & 0 \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{Q}_{\text{total}} \\ -\boldsymbol{\gamma} \end{bmatrix}. \tag{7.5}$$

For five planar bodies there are 15 acceleration unknowns; the 13 multipliers give 28 total unknowns. With positive masses and inertias, M is positive definite. If J has full row rank at the configuration, the ideal KKT matrix is nonsingular; dependent constraint rows or singular configurations require diagnosis or a suitable rank-revealing/reduced treatment.

For comparison with the notebook's symbolic residual, order the multipliers as (λ_1, λ_2) for P_{12}, (λ_3, λ_4) for P_{23}, (λ_5, λ_6) for P_{34}, (λ_7, λ_8) for P_{45}, $(\lambda_9, \lambda_{10})$ for P_{51}, and $(\lambda_{11}, \lambda_{12}, \lambda_{13})$ for the fixed x_4, y_4, θ_4 pose. With $c_i = \cos\theta_i$ and $s_i = \sin\theta_i$, the upper block $M\ddot{\boldsymbol{q}} + J^T\boldsymbol{\lambda} - \boldsymbol{Q}_g = \boldsymbol{0}$ is the following set of 15 scalar body equations:

$$\begin{aligned}
\text{body 1:}\quad & m_1\ddot{x}_1 + \lambda_1 - \lambda_9 = 0,\\
& m_1\ddot{y}_1 + gm_1 - \lambda_{10} + \lambda_2 = 0,\\
& I_1\ddot{\theta}_1 - \tfrac{1}{2}L_1\lambda_1 s_1 + \tfrac{1}{2}L_1\lambda_{10}c_1 + \tfrac{1}{2}L_1\lambda_2 c_1 - \tfrac{1}{2}L_1\lambda_9 s_1 = 0,\\
\text{body 2:}\quad & m_2\ddot{x}_2 - \lambda_1 + \lambda_3 = 0,\\
& m_2\ddot{y}_2 + gm_2 - \lambda_2 + \lambda_4 = 0,\\
& I_2\ddot{\theta}_2 - \tfrac{1}{2}L_2\lambda_1 s_2 + \tfrac{1}{2}L_2\lambda_2 c_2 - \tfrac{1}{2}L_2\lambda_3 s_2 + \tfrac{1}{2}L_2\lambda_4 c_2 = 0,\\
\text{body 3:}\quad & m_3\ddot{x}_3 - \lambda_3 + \lambda_5 = 0,\\
& m_3\ddot{y}_3 + gm_3 - \lambda_4 + \lambda_6 = 0,\\
& I_3\ddot{\theta}_3 - \tfrac{1}{2}L_3\lambda_3 s_3 + \tfrac{1}{2}L_3\lambda_4 c_3 - \tfrac{1}{2}L_3\lambda_5 s_3 + \tfrac{1}{2}L_3\lambda_6 c_3 = 0,\\
\text{body 4:}\quad & m_4\ddot{x}_4 + \lambda_{11} - \lambda_5 + \lambda_7 = 0,\\
& m_4\ddot{y}_4 + gm_4 + \lambda_{12} - \lambda_6 + \lambda_8 = 0,\\
& I_4\ddot{\theta}_4 + \lambda_{13} - \tfrac{1}{2}L_4\lambda_5 s_4 + \tfrac{1}{2}L_4\lambda_6 c_4 - \tfrac{1}{2}L_4\lambda_7 s_4 + \tfrac{1}{2}L_4\lambda_8 c_4 = 0,\\
\text{body 5:}\quad & m_5\ddot{x}_5 - \lambda_7 + \lambda_9 = 0,\\
& m_5\ddot{y}_5 + gm_5 - \lambda_8 + \lambda_{10} = 0,\\
& I_5\ddot{\theta}_5 - \tfrac{1}{2}L_5\lambda_7 s_5 + \tfrac{1}{2}L_5\lambda_8 c_5 - \tfrac{1}{2}L_5\lambda_9 s_5 + \tfrac{1}{2}L_5\lambda_{10} c_5 = 0.
\end{aligned} \tag{7.6}$$

The first two equations for each body are translational force balance; the third is moment balance about its center of mass. The three extra multipliers act only on the grounded link's pose rows. Together these scalar equations are exactly the upper 15 rows of (7.5); the lower 13 rows are the expanded acceleration residuals in Chapter 6.

7.4.1 Symbolic assembly and one constrained solve

The notebook keeps m_i, I_i, L_i, and g symbolic while constructing the model. Its diagonal mass matrix follows the coordinate order (x_i, y_i, θ_i), and gravity contributes only to translational vertical entries. The following algorithm separates exact derivation from numerical evaluation:

1. Create symbolic vectors $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ in body order, and symbolic arrays for five masses, inertias, and link lengths.

2. Assemble $M = \mathrm{diag}(m_1, m_1, I_1, \ldots, m_5, m_5, I_5)$ and $V = \sum_i m_i g y_i$; differentiate to obtain $\boldsymbol{Q}_g = -\partial V / \partial \boldsymbol{q}$.

3. Import $\boldsymbol{\Phi}$ and J from the pin geometry. Form $\boldsymbol{\gamma} = \dot{J}\dot{\boldsymbol{q}}$ by differentiating $J\dot{\boldsymbol{q}}$ and removing its $J\ddot{\boldsymbol{q}}$ contribution.

4. Define 13 multipliers in the same row order as $\boldsymbol{\Phi}$. Assemble the 28-by-28 matrix in (7.5) and its right-hand side $[\boldsymbol{Q}_{\mathrm{total}}^T, -\boldsymbol{\gamma}^T]^T$.

5. Before solving, check $\mathrm{rank}(J) = 13$ at the current pose. Solve simultaneously for 15 accelerations and 13 multipliers; do not eliminate reaction rows by assuming the coordinates are independent.

6. Evaluate the force residual $M\ddot{\boldsymbol{q}} + J^T\boldsymbol{\lambda} - \boldsymbol{Q}_{\mathrm{total}}$ and the ideal acceleration residual $J\ddot{\boldsymbol{q}} + \boldsymbol{\gamma}$. Require both to be small relative to the chosen numerical scales.

In the numerical notebook, the mass, inertia, and link-length substitutions are performed only after the symbolic expressions have been formed. Because a raw infinity norm over all 15 force-balance rows mixes newtons and newton-metres, the initial solve is reported by component family: the maximum translational force residual is $1.998 \times 10^{-15}\,\mathrm{N}$, and the maximum moment residual is $4.663 \times 10^{-15}\,\mathrm{N\,m}$. These check the linear solve at that state; the separate trajectory checks are needed to assess integration behavior.

The multipliers enforce ideal reaction forces; they are not extra physical state coordinates. The full constrained model also includes the algebraic position equation $\boldsymbol{\Phi}(\boldsymbol{q}) = \boldsymbol{0}$, so the dynamics form a DAE. Solving (7.5) at each state evaluation yields accelerations compatible with the acceleration constraints, but numerical integration still needs consistent initial data and may require drift control.

7.5 Ideal equations and numerical stabilization

Equation (7.5) uses the ideal acceleration constraint. Baumgarte stabilization modifies the constraint right-hand side by feeding back position and velocity errors. Such terms belong to the numerical integration formulation, not to the ideal force balance or physical applied loads. Chapter 9 compares that choice with projection, direct DAE integration, coordinate reduction, and penalty approaches.

7.6 Summary

The closed-chain acceleration solve couples rigid-body force balance to acceleration compatibility. Lagrange multipliers encode ideal joint reactions, while the lower KKT block enforces the differentiated loop constraints. The next chapter prepares position and velocity initial data consistent with the constraint manifold.

Chapter 8

Consistent Initial Conditions

8.1 State of the Art

Constrained dynamics requires initial data compatible with its algebraic constraints. Projection corrects approximate positions and velocities toward the constraint manifold, while DAE formulations can impose additional consistency requirements [2, 5]. This focused review covers the geometric and numerical consistency requirements used in the example; it does not compare projection algorithms or DAE packages. Bibliographic scope checked 2026-10-07.

8.2 Position and velocity consistency

For the closed-chain model, the initial position $\boldsymbol{q}_0$ must satisfy

$$\boldsymbol{\Phi}(\boldsymbol{q}_0) = \mathbf{0}. \tag{8.1}$$

The initial velocity must be tangent to the constraint manifold:

$$J(\boldsymbol{q}_0)\dot{\boldsymbol{q}}_0 = \mathbf{0}. \tag{8.2}$$

Thus, a configuration can satisfy all pin positions while a proposed velocity still separates one or more pins. Both checks are needed before integration.

Initial acceleration is not an independent initial datum in this acceleration-level formulation. Once $\boldsymbol{q}_0$ and $\dot{\boldsymbol{q}}_0$ are consistent, solve the equations of motion and acceleration constraint to obtain $\ddot{\boldsymbol{q}}_0$ and the reactions $\boldsymbol{\lambda}_0$.

8.3 Projection of approximate initial data

In practice, geometry may be specified approximately. Starting from a trial position $\boldsymbol{q}_k$, linearize the residual and choose a correction $\Delta\boldsymbol{q}_k$ satisfying

$$J(\boldsymbol{q}_k)\Delta\boldsymbol{q}_k \approx -\boldsymbol{\Phi}(\boldsymbol{q}_k), \qquad \boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \Delta\boldsymbol{q}_k. \tag{8.3}$$

Because the system uses redundant coordinates, the correction is not unique. A least-squares solve can select a minimum-norm correction; repeat the linearization until the position residual meets a stated tolerance or report failure to converge. Check that the Jacobian has the expected rank before interpreting this correction as a regular projection.

After obtaining a consistent position, correct a trial velocity $\dot{\boldsymbol{q}}_{\text{trial}}$ by solving

$$J(\boldsymbol{q}_0)\Delta\dot{\boldsymbol{q}} = -J(\boldsymbol{q}_0)\dot{\boldsymbol{q}}_{\text{trial}}, \qquad \dot{\boldsymbol{q}}_0 = \dot{\boldsymbol{q}}_{\text{trial}} + \Delta\dot{\boldsymbol{q}}. \tag{8.4}$$

Again, a least-squares solution selects one admissible correction, not a unique physical choice. If initial velocities have physical meaning, choose a correction criterion that preserves the intended motion as closely as possible.

8.3.1 Initialization algorithm

The notebook projects only the initial state. Its operation can be summarized as follows:

1. Form a geometric position guess from the specified pins, compute $\boldsymbol{\Phi}(\boldsymbol{q}_{\text{guess}})$ and $J(\boldsymbol{q}_{\text{guess}})$, and verify that J has the expected row rank.
2. While $\|\boldsymbol{\Phi}(\boldsymbol{q}_k)\|_\infty$ exceeds the position tolerance, solve the minimum-norm least-squares correction $J(\boldsymbol{q}_k)\Delta\boldsymbol{q}_k \approx -\boldsymbol{\Phi}(\boldsymbol{q}_k)$ and update $\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \Delta\boldsymbol{q}_k$.
3. Stop with a reported failure if the residual does not reach tolerance within the iteration limit; do not silently pass an inconsistent pose to the integrator.
4. At the corrected position, evaluate J and solve the minimum-norm velocity correction $J\Delta\dot{\boldsymbol{q}} = -J\dot{\boldsymbol{q}}_{\text{trial}}$. Set $\dot{\boldsymbol{q}}_0 = \dot{\boldsymbol{q}}_{\text{trial}} + \Delta\dot{\boldsymbol{q}}$.
5. Check both $\|\boldsymbol{\Phi}(\boldsymbol{q}_0)\|_\infty$ and $\|J(\boldsymbol{q}_0)\dot{\boldsymbol{q}}_0\|_\infty$. Then solve the constrained acceleration equations once and check force-balance and acceleration-constraint residuals before starting the trajectory integration.

For the notebook's projected initial state, the raw pin position residual is $4.441 \times 10^{-16}\,\text{m}$, grounded translation error is zero, and grounded angle error is zero. The pin velocity residual is $4.718\times10^{-16}\,\text{m}\,\text{s}^{-1}$; the grounded translation-rate and angular-rate residuals are $1.561 \times 10^{-17}\,\text{m}\,\text{s}^{-1}$ and $1.648 \times 10^{-17}\,\text{s}^{-1}$, respectively. These values verify that particular initialization; tolerances should be selected relative to units and model scales in other applications.

8.4 Checks before integration

Before starting the solver, evaluate $\|\boldsymbol{\Phi}(\boldsymbol{q}_0)\|_\infty$ and $\|J(\boldsymbol{q}_0)\dot{\boldsymbol{q}}_0\|_\infty$. Confirm the Jacobian rank expected for the selected configuration, then solve the constrained acceleration equations and check both force-balance and acceleration-constraint residuals. Tolerances should be reported with units and interpreted relative to the model scales; there is no universal projection tolerance for every mechanism.

In the five-link notebook, position and velocity projection are performed on the initial guess before integration. Projection is not applied after every time step in that example; subsequent drift is controlled by the selected Baumgarte term and monitored separately.

8.5 Summary and transition

Consistent initialization requires position closure and tangent velocity. Acceleration follows from the constrained equations. Projection offers a practical correction for approximate initial data, but its nonuniqueness and rank assumptions should be stated. With an admissible initial state prepared, the next chapter considers time integration and constraint drift.

Chapter 9

Time Integration and Constraint Drift

9.1 State of the Art

Constrained mechanical models can be integrated as differential-algebraic equations, reduced to independent coordinates, or converted into an ordinary differential equation using differentiated constraints. These choices interact with time-stepping and drift control but are not the same decision. Baumgarte's feedback method modifies the acceleration constraint, while projection methods correct a numerical state back toward the constraint manifold [8, 5]. Structure-preserving integration provides another important perspective for long-time simulation, though it does not by itself select a loop-constraint treatment [12]. This is a focused orientation based on these foundational sources and the SciPy 1.18.1 documentation accessed 2026-10-07, not an exhaustive review of current constrained-mechanics solvers.

9.2 Separate the model from the stepper

The ideal closed-chain equations are

$$M(q)\ddot{q} + J(q)^T\lambda = Q(q, \dot{q}, t), \tag{9.1}$$

$$\Phi(q) = 0. \tag{9.2}$$

They are a differential-algebraic system. Differentiating the constraint twice yields

$$J(q)\ddot{q} = -\dot{J}(q, \dot{q})\dot{q}. \tag{9.3}$$

Solving this acceleration-level saddle-point system at each state evaluation determines $\ddot{q}$ and λ. Returning $[\dot{q}, \ddot{q}]$ then defines an ODE for the state $y = [q, \dot{q}]^T$, but integration error can still move the state away from $\Phi = 0$ and $J\dot{q} = 0$. It is therefore important to report both the chosen differential formulation and the time integrator.

9.3 Drift control

9.3.1 Baumgarte stabilization

Let Φ be the position residual. Baumgarte stabilization prescribes a damped second-order error equation,

$$\ddot{\Phi} + 2\zeta\omega_B\dot{\Phi} + \omega_B^2\Phi = 0, \tag{9.4}$$

where $\zeta > 0$ and $\omega_B > 0$ are numerical gains. Using $\dot{\Phi} = J\dot{q}$ gives the stabilized acceleration constraint

$$J\ddot{q} = -\dot{J}\dot{q} - 2\zeta\omega_B J\dot{q} - \omega_B^2\Phi. \tag{9.5}$$

The last two terms are feedback corrections, not physical forces or part of the ideal model. Increasing ω_B may improve constraint-error decay but also introduce fast time scales and stiffness. Gains must be evaluated with the integration method and residuals, not selected from a universal rule.

9.3.2 Projection

Position projection solves a nonlinear correction problem that returns q toward $\Phi(q) = 0$. A local least-norm correction satisfies

$$J(q_k)\Delta q = -\Phi(q_k), \qquad q_{k+1} = q_k + \Delta q. \tag{9.6}$$

Velocity projection at a corrected position similarly solves $J\Delta\dot{q} = -J\dot{q}$. Projection may be used for initialization or after accepted integration steps. It changes the numerical state and can affect invariants; its frequency and norm should be reported. The notebook projects only the initial position and velocity, not every later state.

9.3.3 Other formulations

Coordinate reduction eliminates dependent coordinates and constraints locally, but requires a suitable parameterization and may become singular or cumbersome as the mechanism changes configuration. Penalty methods replace exact constraints with stiff forces proportional to constraint error; they avoid explicit multipliers but introduce penalty parameters and potentially severe stiffness. A DAE solver treats algebraic constraints as part of the integrated problem and has solver-specific consistency requirements. These are formulation choices; each can be paired with different time discretizations. DAE solver theory and stiff integration are treated by Ascher and Petzold and by Hairer and Wanner [5, 6].

9.4 Time-stepper and tolerances

For the demonstration, the acceleration solve with Baumgarte feedback defines an ODE integrated by SciPy `solve_ivp`. Its default method is RK45, an explicit Runge–Kutta method of order 5(4). Relative and absolute tolerances control local error estimates, while `max_step` bounds accepted step size; `t_eval` specifies stored output times and does not force internal steps at those points [15]. Explicit Runge–Kutta methods are appropriate first choices for non-stiff problems; the documentation recommends implicit methods such as Radau or BDF when stiffness is present. A Baumgarte gain that creates fast correction dynamics may require a stiff solver or a different drift treatment.

For the five-link run, the Baumgarte settings are $\zeta = 1$ and $\omega_B = 20\,\mathrm{s}^{-1}$. The ODE solver uses relative tolerance 10^{-8}, absolute tolerance 10^{-10}, and maximum step 0.01 s over ten seconds. These are reproducible example settings, not a claim of optimality. A numerical study should vary tolerances, maximum step, and stabilization gains and compare constraint errors and quantities of interest.

9.5 Algorithmic outline

1. Project approximate initial positions onto $\Phi(q) = 0$, then project velocities onto $J(q)\dot{q} = 0$.
2. At each state evaluation, compute Φ, J, $\dot{J}\dot{q}$, and the applied generalized forces.
3. Solve the stabilized KKT system for acceleration and multipliers.
4. Return the state derivative formed by the generalized velocity and the solved acceleration.
5. Integrate with an adaptive solver and record status, tolerances, solver evaluations, and saved output times.
6. Independently evaluate position and velocity constraint residuals along the trajectory; adjust the method only after inspecting those diagnostics.

9.5.1 Notebook execution sequence

The canonical five-link notebook implements the above as an ODE right-hand side. Written without language-specific syntax, one evaluation proceeds as follows:

1. Split $y = [q^T, \dot{q}^T]^T$ into the 15 generalized positions and 15 velocities.
2. Evaluate $\Phi(q)$, $J(q)$, $\gamma(q, \dot{q}) = \dot{J}\dot{q}$, and check that J has the expected rank 13. Stop with a diagnostic if the constraints lose rank.
3. Form the stabilized lower-block right-hand side

$$r_c = -\gamma - 2\zeta\omega_B J\dot{q} - \omega_B^2 \Phi. \tag{9.7}$$

4. Solve the block system with M, J^T, J, and a zero lower-right block for $\ddot{q}$ and λ, using $[Q_{\text{total}}^T, r_c^T]^T$ as its right-hand side.
5. Return $\dot{y} = [\dot{q}^T, \ddot{q}^T]^T$. The multipliers are algebraic outputs, not additional integrated state variables.

For the recorded baseline, the initial state is projected once and then integrated for 10 s with RK45, relative tolerance 10^{-8}, absolute tolerance 10^{-10}, and maximum internal step 0.01 s. The solver is asked to save 2,001 evenly spaced output samples; those requested times do not force its internal adaptive steps. After the run, compute $\|\Phi(q_k)\|_\infty$, $\|J(q_k)\dot{q}_k\|_\infty$, and grounded-pose error at every saved state, and record solver status and RHS evaluation count. The notebook then repeats the initial-value problem with tighter tolerances and with ω_B reduced from 20 to $10\,\text{s}^{-1}$, comparing residual maxima and maximum coordinate difference from baseline. These repeats are sensitivity checks, not a formal convergence proof.

9.6 Summary

Constraint enforcement, drift control, and time integration are separate choices. This book's example uses an acceleration-level KKT solve, initial projection, Baumgarte feedback, and adaptive RK45. The next chapter turns these choices into a verification and validation plan.

Chapter 10

Verification and Validation

10.1 State of the Art

Verification asks whether a computational model and its implementation correctly solve the equations they claim to solve; validation asks whether those equations adequately represent the physical system for an intended use. Both require documented evidence, but they answer different questions. A general treatment of verification and validation in scientific computing is given by Oberkampf and Roy [16]; reproducible computational research also depends on recording data, code, parameters, and software context [17]. This focused review uses these foundational sources, checked 2026-10-07; it is a practical checklist, not a survey of domain-specific experimental standards.

10.2 Verification layers

10.2.1 Symbolic and equation checks

Before numerical integration, verify coordinate order, dimensions, force signs, and constraint row order. For the five-link model, the symbolic residual vector includes ten pin equations and three fixed-pose equations. Differentiation produces $J = \partial\Phi/\partial q$ and the acceleration bias $\dot{J}\dot{q}$. These expressions are evaluated at a common state and compared against the independent numerical implementation. The symbolic ideal acceleration constraint and force-balance residuals should simplify or evaluate to zero within arithmetic precision.

The notebook applies these checks in this order:

1. Keep m_i, I_i, L_i, g symbolic while constructing T, V, Φ, J, and $\dot{J}\dot{q}$; verify that the expressions contain no floating-point coefficients and that each L_i remains present in the relevant geometry.

2. Check dimensions: $q, \dot{q}, \ddot{q} \in \mathbb{R}^{15}$, $\Phi, \lambda, \dot{J}\dot{q} \in \mathbb{R}^{13}$, and the KKT matrix is 28×28.

3. At one common projected state, substitute numerical values into the symbolic model and compare Φ, J, and $\dot{J}\dot{q}$ elementwise with the separately written numerical geometry routines.

4. Solve the KKT system and compute $r_f = M\ddot{q} + J^T\lambda - Q$ and $r_a = J\ddot{q} + \dot{J}\dot{q}$ for the ideal system, or the matching stabilized residual when stabilization is enabled. Report translational force rows in newtons separately from rotational moment rows in newton-metres.

For this derivation the symbolic model reports 15 coordinates, symbolic lengths $L_1, \ldots, L_5$, ten pin rows, and three fixed-pose rows. Its exact-symbol check confirms that no floating-point coefficients have entered Φ, J, M, T, V, or Q_g; the ideal 28-by-28 KKT system contains no Baumgarte gains, which occur only in the separate stabilized constraint. These checks ensure numerical geometry has not accidentally been baked into the symbolic equations. For the notebook's initial-state comparison, the maximum symbolic/numeric differences are 3.331×10^{-16} for Φ, 2.220×10^{-16} for J, and 4.337×10^{-19} for $\dot{J}\dot{q}$. These values document that implementation pair at that pose.

10.2.2 Linear solve checks

At each acceleration solve, check the KKT residual

$$r_{\mathrm{KKT}} = \begin{bmatrix} M & J^T \\ J & 0 \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \end{bmatrix} - \begin{bmatrix} Q \\ -\dot{J}\dot{q} - 2\zeta\omega_B J\dot{q} - \omega_B^2 \Phi \end{bmatrix}. \tag{10.1}$$

For the upper KKT block, report translational force and rotational moment residuals separately in newtons and newton-metres, or scale each row by a documented characteristic force or moment before combining them. For the lower block, report pin and grounded-translation acceleration residuals in metres per second squared separately from the grounded angular-acceleration residual in radians per second squared. Likewise, Φ contains pin-position residuals in metres and a fixed-angle residual in radians, while $J\dot{q}$ contains translational speeds and angular rates; report pin and ground components separately or use explicit scales. Confirm that J has the expected rank at the tested pose. A successful factorization alone is not evidence that equations, signs, or units are correct.

10.2.3 Trajectory checks

Because the constraint vector contains both translational and angular rows, report separate component norms:

$$\begin{aligned} e_{p,\mathrm{pin}} &= \max_{1 \le j \le 10} |\Phi_j|, & e_{p,\mathrm{ground}} &= \max(|\Phi_{11}|, |\Phi_{12}|), & e_{\theta,\mathrm{ground}} &= |\Phi_{13}|, \\ e_{v,\mathrm{pin}} &= \max_{1 \le j \le 10} |(J\dot{q})_j|, & e_{v,\mathrm{ground}} &= \max(|(J\dot{q})_{11}|, |(J\dot{q})_{12}|), & e_{\omega,\mathrm{ground}} &= |(J\dot{q})_{13}|. \end{aligned} \tag{10.2}$$

The pin position and velocity norms have units of metres and metres per second; the ground translation and translation-rate norms have the same respective units, while the ground angle and angular-rate norms are in radians and radians per second. The trajectory check loops over every saved state, reevaluates $\Phi(q_k)$ and $J(q_k)$, and records the maxima of these components. It also confirms that the solver reports success, reaches the requested final time, and returns finite state values. Repeat with tighter tolerances or a smaller maximum step and compare the trajectory and quantities relevant to the intended use. Output sampling density alone is not an integration convergence study.

10.2.4 Validation evidence

The current five-link model is an ideal planar rigid-body mechanism. Its notebook verifies internal consistency and numerical closure but does not compare predictions against a physical prototype or measurement data. It therefore demonstrates code verification and numerical consistency, not empirical validation. A validation study would additionally need measured geometry, masses/inertias, joint behavior, loading, sensor uncertainty, and comparison metrics appropriate to the intended application [16].

10.3 Acceptance checks for the examples

Evidence	Check	Five-link result
Initial consistency	Pin position and velocity residuals after projection	$4.441 \times 10^{-16}\,\mathrm{m}$; $4.718 \times 10^{-16}\,\mathrm{m\,s^{-1}}$
Initial grounding	Translation, angle, translation-rate, angular-rate residuals	$0\,\mathrm{m}$; $0\,\mathrm{rad}$; $1.561 \times 10^{-17}\,\mathrm{m\,s^{-1}}$; $1.648 \times 10^{-17}\,\mathrm{s^{-1}}$
Local mobility	Rank of J at the initial pose	Rank 13; two remaining DOF
Equation implementation	Symbolic/numeric Φ, J, and bias comparison	Differences below 4×10^{-16}
Force solve	Initial KKT residual by component family	$1.998 \times 10^{-15}\,\mathrm{N}$ force; $4.663 \times 10^{-15}\,\mathrm{N\,m}$ moment
Integration	Solver reaches 10 s successfully	2,001 saved samples
Constraint preservation	Maximum pin position and velocity errors	$1.108 \times 10^{-9}\,\mathrm{m}$; $1.247 \times 10^{-7}\,\mathrm{m\,s^{-1}}$
Grounding	Translation, angle, translation-rate, angular-rate maxima	$1.333 \times 10^{-17}\,\mathrm{m}$; $3.031 \times 10^{-19}\,\mathrm{rad}$; $1.092 \times 10^{-15}\,\mathrm{m\,s^{-1}}$; $1.648 \times 10^{-17}\,\mathrm{s^{-1}}$
Energy behavior	Relative mechanical-energy drift in the baseline run	3.158×10^{-7}

These values describe one parameter set, one solver configuration, and sampled output states. They do not guarantee accuracy between samples, validate the idealized model against a physical mechanism, or establish robustness near singular configurations.

10.4 Algorithmic outline

1. Verify symbolic expressions and their dimensions before assigning numerical parameters.
2. Compare symbolic and independently coded kinematics, Jacobians, and acceleration bias at multiple admissible states.
3. Check rank and KKT residuals at representative configurations.
4. Integrate from projected initial data and inspect solver success and state finiteness.
5. Evaluate position, velocity, and fixed-pose errors at every saved state.
6. Refine tolerances and step limits, vary stabilization gains, and report trajectory and residual sensitivity rather than relying on a single run.
7. Label any comparison with analytical solutions, benchmarks, or measurements according to the evidence actually available.

10.5 Summary

Symbolic identities, independent implementation comparisons, linear residuals, trajectory constraints, and refinement studies provide complementary verification evidence. Physical validation requires experimental or otherwise independent evidence not present in this example.

Chapter 11

Results and Visualization

11.1 State of the Art

Simulation results should report model parameters, solver settings, diagnostics, and enough implementation detail for reproduction. Reproducible-computation guidance emphasizes preserving the code and computational context behind reported results [17]. This focused review uses that reproducibility guidance, checked 2026-10-07. The chapter reports one run from the canonical five-link notebook; it is not a parameter study, benchmark comparison, or experimental validation.

11.2 Example configuration

The model contains five planar rigid bodies with coordinates ordered (x_i, y_i, θ_i). The five perimeter pins and a fixed pose for link 4 give 13 independent constraints at the initial pose, leaving two local degrees of freedom. The masses are $(1.2, 0.9, 1.1, 1.0, 0.8)\,\mathrm{kg}$, and the planar inertias are $(0.30, 0.22, 0.28, 0.25, 0.18)\,\mathrm{kg\,m^2}$. Derived center-to-center link lengths are

$$(3.19061,\ 1.08167,\ 2.05913,\ 1.00000,\ 1.86815)\,\mathrm{m}.$$

Gravity is $9.81\,\mathrm{m\,s^{-2}}$. The links are ideal, rigid, and unactuated; link 4 is grounded.

The initial generalized velocity is constructed by perturbing the angular velocity of link 1 and projecting onto the velocity constraints. The ODE run uses the settings listed in Chapter 9. No damping, friction, contact, or external drives are included.

11.3 Numerical result

The projected initial state has pin position residual $4.441\times10^{-16}\,\mathrm{m}$ and pin velocity residual $4.718\times10^{-16}\,\mathrm{m\,s^{-1}}$. The grounded translation and angle residuals are zero, while their rates are $1.561\times10^{-17}\,\mathrm{m\,s^{-1}}$ and $1.648\times10^{-17}\,\mathrm{s^{-1}}$. The initial constraint Jacobian has rank 13. At the initial KKT solve, the maximum translational force residual is $1.998\times10^{-15}\,\mathrm{N}$, the maximum moment residual is $4.663\times10^{-15}\,\mathrm{N\,m}$, and the maximum stabilized pin-acceleration residual is $7.834\times10^{-15}\,\mathrm{m\,s^{-2}}$. The 10 s solve completes successfully with 2,001 saved output states and 13,154 right-hand-side evaluations. Across saved states, the maximum pin-position closure error is $1.108\times10^{-9}\,\mathrm{m}$, and maximum pin-velocity residual is $1.247\times10^{-7}\,\mathrm{m\,s^{-1}}$. Link 4's maximum translation error is $1.333\times10^{-17}\,\mathrm{m}$, its maximum angle error is $3.031\times10^{-19}\,\mathrm{rad}$, and its final pin position residual is $9.353\times10^{-11}\,\mathrm{m}$.

Relative mechanical-energy drift, scaled by the larger of one joule or the initial energy magnitude, is 3.158×10^{-7} for this Baumgarte-stabilized run.

The initial and final configurations show substantial motion of the unfixed links while the grounded pose remains fixed. The outline is drawn by connecting joint attachment points and is only a visualization proxy: the equations use full rigid-body masses and inertias. The animation samples saved solver output for display; it does not alter the integration or establish accuracy by itself.

11.3.1 Figure and animation generation

The notebook builds the static configuration figure and movie from the saved generalized-coordinate history. Its plotting procedure is:

1. For each selected saved state, read every body's center and orientation and map its local pin offsets into world coordinates.
2. Plot the five pin-to-pin outlines and center markers. If a filled body outline is shown, sort the attachment points by their polar angle about the center and close the polygon; label this as a display proxy, not the rigid-body geometry used in the equations.
3. For the initial/final comparison, draw the projected initial state and the last saved state on equal-scale axes.
4. For the movie, compute world bounds from all saved centers and pins over the full trajectory, then hold those axes fixed. Select evenly spaced saved states at the chosen display frame rate and update existing line, marker, label, pin, and time artists for each frame.
5. Encode the frames, reopen the written animation, and verify both its frame count and the sum of encoded frame durations against the intended frame rate and simulation duration.

In the notebook, the ten-second run produces 200 frames at 20 frames per second. This verifies the movie artifact's timing and frame count; the plotted path remains a visualization of the numerical result, not a substitute for constraint, solver, or sensitivity checks.

11.4 Sensitivity to tolerances and stabilization

Two additional ten-second runs keep the initial state and output times fixed. Tightening the relative and absolute tolerances from $10^{-8}, 10^{-10}$ to $10^{-9}, 10^{-11}$, while keeping the maximum step at 0.01 s and $\omega_B = 20\,\mathrm{s}^{-1}$, increases RHS evaluations from 13,154 to 20,492. The maximum pin-velocity residual falls from 1.247×10^{-7} to $1.019 \times 10^{-8}\,\mathrm{m\,s}^{-1}$, while maximum pin-position closure is 9.746×10^{-11} m. Relative energy drift falls to 2.548×10^{-8}. The maximum coordinate difference from the baseline sampled trajectory is 9.397×10^{-5}.

Halving the Baumgarte frequency to $10\,\mathrm{s}^{-1}$ at baseline solver tolerances reduces RHS evaluations to 8,654. Maximum pin-position and pin-velocity residuals become 2.018×10^{-9} m and $1.058 \times 10^{-7}\,\mathrm{m\,s}^{-1}$; relative energy drift is 3.191×10^{-7}, and the maximum coordinate difference from baseline is 2.405×10^{-4}. The Jacobian rank remains 13 at every saved state in both runs. In both sensitivity runs, grounded-pose translation stays below 4.4×10^{-17} m and grounded-angle error below 6.1×10^{-19} rad. These comparisons show a tradeoff between computational work, constraint residuals, and trajectory sensitivity. They are a small sensitivity check, not a convergence proof or a broad study of solver/gain combinations.

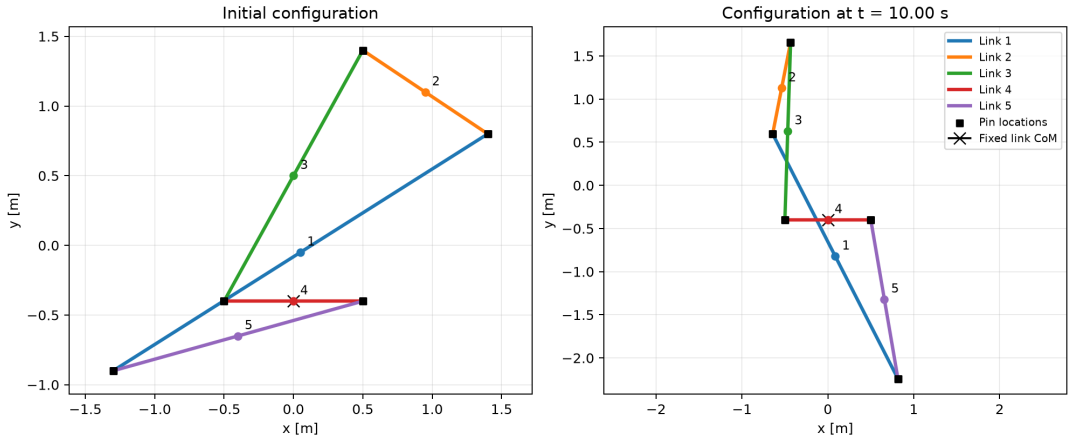


Figure 11.1: Initial and final configurations from the 10 s five-link run. Lines join pin attachment points to visualize rigid-link orientation; they do not depict body deformation.

11.5 Reproducibility record

The canonical executable artifact is the five-link dynamics notebook in the project root. The recorded environment uses Python from the project virtual environment with NumPy 2.5.3, SymPy 1.14.0, SciPy 1.18.1, Matplotlib 3.11.2, and Pillow 12.3.0. The notebook regenerates the 10 s, 200-frame GIF and checks its encoded duration. The result is deterministic only to the extent that numerical library versions, solver settings, and the platform remain equivalent; figures and metrics should be regenerated after changes.

11.6 Summary

This run demonstrates a moving closed-chain solution with small sampled position and velocity residuals under one explicit set of parameters and numerical settings. The values are evidence for this run, not universal accuracy thresholds. Chapter 12 bounds what can be inferred and identifies extensions needed for higher-fidelity applications.

Chapter 12

Limitations and Extensions

12.1 State of the Art

Multibody simulation spans rigid and flexible bodies, ideal and compliant joints, open and closed topologies, and models with contact, friction, and actuation. The appropriate fidelity depends on the engineering question; increasing model detail can introduce additional states, parameters, nonsmooth events, and numerical stiffness. Shabana provides broader coverage of rigid and deformable multibody formulations, while differential-algebraic and numerical-integration references describe important solver consequences [2, 5, 6]. This focused scope-setting synthesis uses these foundational works, checked 2026-10-07; it is not a survey of each extension area.

12.2 Limitations of the worked model

The running mechanism is planar and consists of rigid links with constant center-of-mass coordinates, constant mass properties, ideal revolute pins, and a perfectly grounded fourth link. It omits:

- joint clearance, friction, backlash, and compliance;
- link flexibility, structural damping, and material limits;
- impacts, unilateral contact, collision detection, and frictional contact;
- actuator dynamics, control laws, and time-varying external loads;
- parameter uncertainty and comparison with measured motion or reaction forces.

The numerical trajectory is one unforced example under gravity with Baumgarte stabilization. Low closure residuals do not prove the physical assumptions are adequate, nor do they guarantee solver robustness at singular configurations.

12.3 Possible extensions

12.3.1 Spatial motion

Three-dimensional bodies require six configuration variables per body in a suitable representation. Rotations introduce additional kinematic structure and, for some parameterizations, coordinate singularities. Joint constraints and mobility must be reformulated for spatial joints rather than obtained by simply copying planar equations.

12.3.2 Flexible bodies and compliant joints

Flexible-body models add elastic coordinates or continuum approximations and require consistent coupling between rigid motion and deformation. Compliant joints replace exact closure conditions with force laws; their parameters should be identified or bounded, and their added high-frequency dynamics may require stiff integration.

12.3.3 Contact, friction, and actuation

Unilateral contact introduces inequality conditions and impact events; friction may make the equations nonsmooth. These cases generally need contact-aware formulations and event or complementarity treatment rather than the smooth ODE used here. Actuation and control can be incorporated through generalized forces, provided the force application points, torque signs, and power balance are stated explicitly.

12.3.4 Robust numerics and model evidence

Future work can compare a dedicated DAE approach, projection after accepted steps, coordinate reduction, and alternative stabilization gains using identical initial data and common error measures. Singular poses, conditioning, energy and momentum behavior, and sensitivity to parameters should be studied. Physical validation additionally requires measurements and uncertainty analysis; it cannot be inferred from symbolic agreement or numerical closure alone.

12.4 Summary

The five-link example isolates the mechanics of planar ideal loop closure but is not a universal model of a real mechanism. Extensions should be selected from the intended use, supported with appropriate physical data, and paired with numerical methods suited to the resulting equations.

Appendix A

Python Implementations

The listings in this appendix give the numerical kernels used by the worked examples. The notebooks contain the complete symbolic derivations, parameter definitions, residual checks, plots, and run instructions; they are the canonical executable artifacts. The snippets below are intentionally limited to the repeated solve steps, so the derivation and validation context is not buried in a long code transcription.

A.1 Double-pendulum acceleration

The notebook derives callable forms of the mass matrix and the vector h from the symbolic Euler–Lagrange residual. The numerical right-hand side solves the linear system directly rather than forming an inverse.

Listing A.1: First-order right-hand side for the unforced double pendulum

```
import numpy as np

def double_pendulum_rhs(time, state, mass_matrix, bias):
        q = state[:2]
        qdot = state[2:]
        matrix = mass_matrix(q)
        h = bias(q, qdot)
        qddot = np.linalg.solve(matrix, -h)
        return np.concatenate((qdot, qddot))
```

For applied generalized torques, replace `-h` by `torque(time, q, qdot) - h`. The notebook's symbolic check verifies the decomposition before the numerical functions are created.

A.2 Five-link constrained acceleration

For the five-link model, the mass matrix is diagonal in center-of-mass coordinates. The constraint model supplies Φ, J, and $\dot{J}\dot{q}$. The following kernel adds Baumgarte feedback only to the lower KKT block; set both gains to zero to recover the ideal acceleration constraint at a consistent state.

Listing A.2: KKT acceleration and reaction-force solve

```
import numpy as np

def constrained_acceleration(q, qdot, time, model,
                                                    zeta=1.0, omega=20.0):
```

```
        phi, jac = model.constraints_and_jacobian(q)
        bias = model.jdot_qdot(q, qdot)
        stabilization = (
                2.0 * zeta * omega * (jac @ qdot)
                + omega**2 * phi
        )
        constraint_rhs = -bias - stabilization

        count = jac.shape[0]
        zero = np.zeros((count, count))
        kkt = np.block([
                [model.mass_matrix, jac.T],
                [jac, zero],
        ])
        rhs = np.concatenate((
                model.generalized_forces(q, qdot, time),
                constraint_rhs,
        ))
        solution = np.linalg.solve(kkt, rhs)
        split = q.size
        return solution[:split], solution[split:]
```

The multiplier signs correspond to the chapter convention $\Phi_{ab} = r_a - r_b$ and $M\ddot{q} + J^T\lambda = Q$. A production implementation should also report Jacobian rank and scaled linear-solve residuals and should handle rank loss explicitly, as the canonical notebook does.

A.3 State derivative and integration

The stabilized acceleration kernel defines an ODE for $y = [q, \dot{q}]^T$. The code below shows the integrator call used by the five-link example; initial position and velocity projection are performed before this call.

Listing A.3: State derivative and adaptive integration

```
from scipy.integrate import solve_ivp

def state_derivative(time, state, model, coordinate_count):
        q = state[:coordinate_count]
        qdot = state[coordinate_count:]
        qddot, _ = constrained_acceleration(
                q, qdot, time, model
        )
        return np.concatenate((qdot, qddot))

time_samples = np.linspace(0.0, 10.0, 2001)
initial_state = np.concatenate((q0, qdot0))
solution = solve_ivp(
        lambda time, state: state_derivative(
                time, state, system, q0.size
        ),
        (0.0, 10.0),
        initial_state,
        t_eval=time_samples,
        rtol=1e-8,
        atol=1e-10,
        max_step=0.01,
```

```
)
if not solution.success:
        raise RuntimeError(solution.message)
```

After integration, evaluate $\|\Phi(q)\|_\infty$, $\|J(q)\dot{q}\|_\infty$, and the grounded-pose error at each saved state. Solver success alone does not establish constraint preservation or physical validity.

Bibliography

[1] Werner Schiehlen. Multibody system dynamics: Roots and perspectives. *Multibody System Dynamics*, 1(2):149–188, 1997.

[2] Ahmed A. Shabana. *Dynamics of Multibody Systems*. Cambridge University Press, 4 edition, 2013.

[3] Andr
'e Laulusa and Olivier A. Bauchau. Review of classical approaches for constraint enforcement in multibody systems. *Journal of Computational and Nonlinear Dynamics*, 3(1):011004, 2008.

[4] Olivier A. Bauchau and Andr
'e Laulusa. Review of contemporary approaches for constraint enforcement in multibody systems. *Journal of Computational and Nonlinear Dynamics*, 3(1):011005, 2008.

[5] Uri M. Ascher and Linda R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Society for Industrial and Applied Mathematics, Philadelphia, 1998.

[6] Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, volume 14 of *Springer Series in Computational Mathematics*. Springer Berlin Heidelberg, 2 edition, 1996.

[7] Martin Arnold. Dae aspects of multibody system dynamics. In *Surveys in Differential-Algebraic Equations IV*, Differential-Algebraic Equations Forum, pages 41–106. Springer International Publishing, Cham, 2017.

[8] J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 1(1):1–16, 1972.

[9] Juan C. Garc
'ia Orden. Energy considerations for the stabilization of constrained mechanical systems with velocity projection. *Nonlinear Dynamics*, 60(1–2):49–62, 2010.

[10] David J. Braun and Michael Goldfarb. Eliminating constraint drift in the numerical simulation of constrained dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 198(37–40):3151–3160, 2009.

[11] Martin Arnold, O. Br
"uls, and Alberto Cardona. Error analysis of generalized-alpha lie group time integration methods for constrained mechanical systems. *Numerische Mathematik*, 129:149–179, 2015.

[12] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer-Verlag, 2 edition, 2006.

[13] Martin Arnold, Bernhard Burgermeister, Claus F
"uhrer, Gerhard Hippmann, and Georg Rill. Numerical methods in vehicle system dynamics: State of the art and current developments. *Vehicle System Dynamics*, 49:1159–1207, 2011.

[14] Ahmed A. Shabana. Flexible multibody dynamics: Review of past and recent developments. *Multibody System Dynamics*, 1(2):189–222, 1997.

[15] The SciPy Community. *scipy.integrate.solve_ivp reference documentation*. Accessed 2026-10-07.

[16] William L. Oberkampf and Christopher J. Roy. *Verification and Validation in Scientific Computing*. Cambridge University Press, 2010.

[17] Geir K. Sandve, Anton Nekrutenko, James Taylor, and Eivind Hovig. Ten simple rules for reproducible computational research. *PLoS Computational Biology*, 9(10):e1003285, 2013.

[18] Herbert Goldstein, Charles P. Poole, and John L. Safko. *Classical Mechanics*. Addison Wesley, San Francisco, 3 edition, 2002.

[19] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, Amit Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Martin J. Curry, Andrew R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sanchit Kulal, Robert Cimrman, and Anthony Scopatz. SymPy: symbolic computing in Python. *PeerJ Computer Science*, 3:e103, 2017.