B

Numerical Methods

From a physical point of view, marine craft kinematics and kinetics are most naturally derived in the continuous-time domain using *Newtonian* or *Lagrangian* dynamics. In the implementation of a control law, it is desirable to represent the nonlinear dynamics in discrete time. This chapter discusses methods for discretization of linear and nonlinear systems, numerical integration and differentiation.

B.1 Discretization of Continuous-Time Systems

This section discusses discretization of linear state-space models with extensions to nonlinear systems using the method of Smith (1977).

Forward Shift Operator

For notational simplicity, let $t_k = kt$ such that $\boldsymbol{x}(k) = \boldsymbol{x}(t_k)$ and $\boldsymbol{x}(k+1) = \boldsymbol{x}(t_k + h)$ where h is the sampling interval. The *forward shift operator* z is defined by

$$\boldsymbol{x}(k+1) := z\boldsymbol{x}(k) \tag{B.1}$$

B.1.1 Linear State-Space Models

Consider the linear continuous-time model

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{B.2}$$

Assume that $\boldsymbol{u}$ is piecewise constant over the sampling interval h and equal to $\boldsymbol{u}(k)$. Hence, the solution of (B.2) can be written

$$\boldsymbol{x}(k+1) = \exp(\boldsymbol{A}h)\boldsymbol{x}(k) + \int_{kh}^{(k+1)h} \exp(\boldsymbol{A}[(k+1)h - \tau])\boldsymbol{B}u(k)\mathrm{d}\tau \tag{B.3}$$

which after integration yields the linear discrete-time model

Handbook of Marine Craft Hydrodynamics and Motion Control, First Edition. Thor I. Fossen.
 ISBN: 978-1-119-99149-6

$$\boldsymbol{x}(k+1) = \boldsymbol{\Phi}\boldsymbol{x}(k) + \boldsymbol{\Delta}\boldsymbol{u}(k) \tag{B.4}$$

where

$$\boldsymbol{\Phi} = \exp(\boldsymbol{A}h), \quad \boldsymbol{\Delta} = \boldsymbol{A}^{-1}(\boldsymbol{\Phi} - \boldsymbol{I})\boldsymbol{B} \tag{B.5}$$

Matlab
The matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Delta}$ can be computed in Matlab as

```
[PHI,DELTA]=c2d(A,B,h)
```

Example B.1 (Discretization of a First-Order Linear System)
Consider the SISO linear system

$$\dot{x} = ax + bu \tag{B.6}$$

$$y = cx + du \tag{B.7}$$

Application of (B.3) yields

$$x(k+1) = \exp(ah)x(k) + \frac{b}{a}[\exp(ah) - 1]u(k) \tag{B.8}$$

$$y(k) = cx(k) + du(k) \tag{B.9}$$

Computation of the Transition Matrix

The transition matrix $\boldsymbol{\Phi}$ can be computed numerically as

$$\boldsymbol{\Phi} = \exp(\boldsymbol{A}h) = \boldsymbol{I} + \boldsymbol{A}h + \frac{1}{2!}\boldsymbol{A}^2h^2 + \cdots + \frac{1}{n!}\boldsymbol{A}^nh^n + \cdots \tag{B.10}$$

Hence,

$$\boldsymbol{\Delta} = \boldsymbol{A}^{-1}(\boldsymbol{\Phi} - \boldsymbol{I})\boldsymbol{B} = h + \frac{1}{2!}\boldsymbol{A}h^2 + \cdots + \frac{1}{n!}\boldsymbol{A}^{n-1}h^n + \cdots \tag{B.11}$$

Consequently, a first-order approximation (Euler discretization) is obtained by

$$\boldsymbol{\Phi} \approx \boldsymbol{I} + \boldsymbol{A}h, \quad \boldsymbol{\Delta} \approx \boldsymbol{B}h \tag{B.12}$$

Alternately, $\boldsymbol{\Phi}$ can be computed by applying a *similarity transformation*

$$\boldsymbol{\Phi} = \exp(\boldsymbol{A}h) = \boldsymbol{E}\exp(\boldsymbol{\Lambda}h)\boldsymbol{E}^{-1} \tag{B.13}$$

where

$$\exp(\boldsymbol{\Lambda} h) = \operatorname{diag}\{\exp(\lambda_i h)\} \tag{B.14}$$

is a diagonal matrix containing the eigenvalues λ_i of $\boldsymbol{A}$ and $\boldsymbol{E}$ is the corresponding eigenvector matrix.

Matlab
The transition matrix can be computed in Matlab as

```
[L,E]=eig(A)
PHI=E*exp(L*h)*inv(E)
```

B.1.2 Nonlinear State-Space Models

Consider the nonlinear model

$$\boldsymbol{M}\ \dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\ \boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\ \boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{B}\boldsymbol{u} \tag{B.15}$$

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}_{\Theta}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{B.16}$$

which can be expressed as a nonlinear time-invariant system

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{B.17}$$

where $\boldsymbol{x} = [\boldsymbol{\eta}^\top, \boldsymbol{\nu}^\top]^\top$ and

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \boldsymbol{J}_{\Theta}(\boldsymbol{\eta})\boldsymbol{\nu} \\ \boldsymbol{M}^{-1}[\boldsymbol{B}\boldsymbol{u} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{g}(\boldsymbol{\eta})] \end{bmatrix} \tag{B.18}$$

Differentiating (B.17) with respect to time yields

$$\ddot{\boldsymbol{x}} = \frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} + \frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}}\dot{\boldsymbol{u}} \tag{B.19}$$

The effect of a zero-order hold in the digital-to-analog converter makes $\dot{\boldsymbol{u}} = \boldsymbol{0}$ over the discrete-time interval. Furthermore, the definition of the Jacobian

$$\boldsymbol{\mathcal{J}}(\boldsymbol{x}) := \frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}} \tag{B.20}$$

implies that the nonlinear continuous equation (B.19) is reduced to a homogeneous equation

$$\ddot{\boldsymbol{x}} = \boldsymbol{\mathcal{J}}(\boldsymbol{x})\dot{\boldsymbol{x}} \tag{B.21}$$

Let

$$\boldsymbol{\mathcal{J}}(\boldsymbol{x}(k)) = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}(k)} \tag{B.22}$$

Hence, the solution of the homogeneous differential equation is

$$\dot{\boldsymbol{x}} = \exp[\boldsymbol{\mathcal{J}}(\boldsymbol{x}(0))(t - t_0)]\ \dot{\boldsymbol{x}}(0) \tag{B.23}$$

Integration of this expression over a sampling interval h finally yields

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \int_0^h \exp[\boldsymbol{\mathcal{J}}(\boldsymbol{x}(k))\tau]\ \dot{\boldsymbol{x}}(k)\ \mathrm{d}\tau \tag{B.24}$$

Example B.2 (Discretization of a Second-Order Nonlinear System)
Consider the SISO nonlinear system

$$\dot{x}_1 = x_2 \tag{B.25}$$

$$\dot{x}_2 = f(x_2) + u \tag{B.26}$$

where $\boldsymbol{x} = [x_1, x_2]^\top$ *is the state vector and u is the input. The Jacobian is found as*

$$\boldsymbol{\mathcal{J}}(\boldsymbol{x}) = \begin{bmatrix} 0 & 1 \\ 0 & a(x_2) \end{bmatrix}, \quad a(x_2) = \frac{\partial f(x_2)}{\partial x_2} \tag{B.27}$$

Hence, applying a similarity transformation:

$$\exp[\boldsymbol{\mathcal{J}}(\boldsymbol{x}(k))t] = \boldsymbol{E}^{-1} \exp(\boldsymbol{\Lambda} t)\boldsymbol{E} \tag{B.28}$$

where $\boldsymbol{\Lambda}$ *is a diagonal matrix containing the eigenvectors of* $\boldsymbol{\mathcal{J}}$ *and* $\boldsymbol{E}$ *is a matrix formed by the corresponding eigenvectors, yields*

$$\exp[\boldsymbol{J}(\boldsymbol{x}(k))t] = \begin{bmatrix} 1 & \frac{1}{a_k}[1 - \exp(a_k t)] \\ 0 & \exp(a_k t) \end{bmatrix} \tag{B.29}$$

where $a_k = a(x_2(k))$*. Hence,*

$$\begin{bmatrix} \boldsymbol{x}_1(k+1) \\ \boldsymbol{x}_2(k+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_1(k) \\ \boldsymbol{x}_2(k) \end{bmatrix} + \int_0^h \begin{bmatrix} 1 & \frac{1}{a_k}[1 - \exp(a_k \tau)] \\ 0 & \exp(a_k \tau) \end{bmatrix} \begin{bmatrix} x_2(k) \\ f(x_2(k)) + u(k) \end{bmatrix} \mathrm{d}\tau \tag{B.30}$$

The discrete model (B.24) can be simplified by approximating the exponential function to the first order, that is

$$\exp[\boldsymbol{\mathcal{J}}(\boldsymbol{x}(k))h] = \boldsymbol{I} + \boldsymbol{\mathcal{J}}(\boldsymbol{x}(k))h + O(h^2) \tag{B.31}$$

B.2 Numerical Integration Methods

In this section numerical solutions to the nonlinear time-varying system

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) \tag{B.32}$$

where the control input $\boldsymbol{u}$ is assumed to be constant over the sampling interval h (zero-order hold), are discussed. Four different methods will be presented.

B.2.1 Euler's Method

A frequently used method for numerical integration is forward Euler:

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + h\,\boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k), t_k) \tag{B.33}$$

The global truncation error for Euler's method is of order $O(h)$.

Applying Euler's method to a second-order system

$$\dot{x} = v \tag{B.34}$$

$$m\dot{v} + dv + kx = \tau \tag{B.35}$$

yields

$$v(k+1) = v(k) + h\left[\frac{1}{m}\tau(k) - \frac{d}{m}v(k) - \frac{k}{m}x(k)\right] \tag{B.36}$$

$$x(k+1) = x(k) + hv(k) \tag{B.37}$$

It should be noted that Euler's method should only be applied to a well-damped second-order system and not an undamped oscillator. In fact an undamped oscillator will yield an unstable solution, as seen from Figure B.1, where the circle in the upper left-hand plot represents the stable region. An undamped oscillator will have eigenvalues on the imaginary axis, which clearly lie outside the circle.

Forward and Backward Euler Integration

A stable method for the undamped second-order system can be obtained by combining the *forward* and *backward* methods of Euler (dotted line in the upper left-hand plot in Figure B.1) according to

$$\text{Forward Euler:} \qquad v(k+1) = v(k) + h\left[\frac{1}{m}\tau(k) - \frac{d}{m}v(k) - \frac{k}{m}x(k)\right] \tag{B.38}$$

$$\text{Backward Euler:} \qquad x(k+1) = x(k) + hv(k+1) \tag{B.39}$$

Extension to Nonlinear Systems

The methods of Euler can be extended to the more general nonlinear system

$$\dot{\boldsymbol{\nu}} = \boldsymbol{M}^{-1}\left[\boldsymbol{B}\boldsymbol{u} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{g}(\boldsymbol{\eta})\right] \tag{B.40}$$

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}_{\Theta}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{B.41}$$

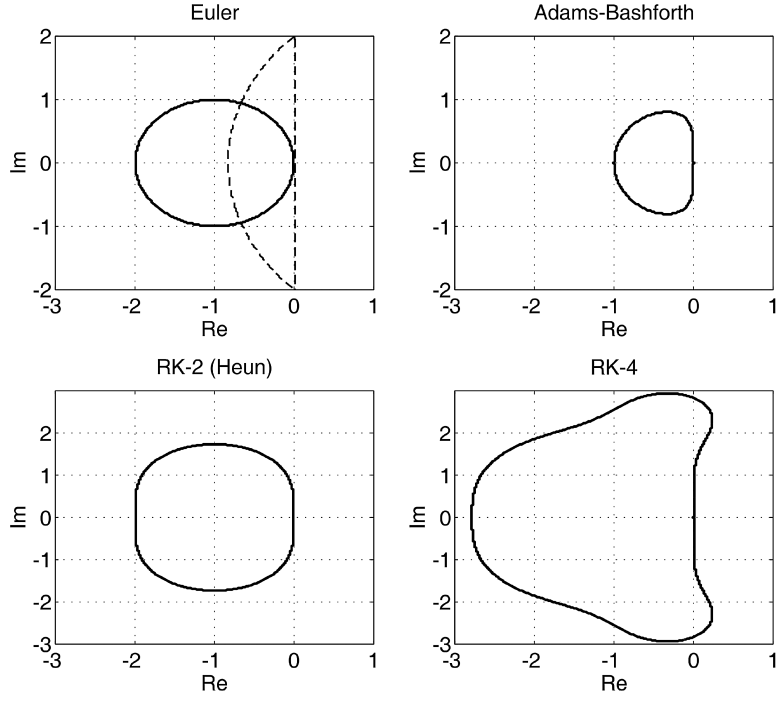

Figure B.1 Stability regions for the Euler, Adams–Bashford, RK-2 and RK-4 methods.

by the following set of discrete-time equations:

$$\boldsymbol{v}(k+1) = \boldsymbol{v}(k) + h\boldsymbol{M}^{-1}\left[\boldsymbol{B}\boldsymbol{u}(k) - \boldsymbol{C}(\boldsymbol{v}(k))\boldsymbol{v}(k) - \boldsymbol{D}(\boldsymbol{v}(k))\boldsymbol{v}(k) - \boldsymbol{g}(\boldsymbol{\eta}(k))\right] \tag{B.42}$$

$$\boldsymbol{\eta}(k+1) = \boldsymbol{\eta}(k) + h\left[\boldsymbol{J}_{\Theta}(\boldsymbol{\eta}(k))\boldsymbol{v}(k+1)\right] \tag{B.43}$$

B.2.2 Adams–Bashford's Second-Order Method

Adams–Bashford integration is more computationally intensive than the schemes of Euler. For instance, the two-step Adams–Bashford integration

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + h\left[\frac{3}{2}\boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k), t_k) - \frac{1}{2}\boldsymbol{f}(\boldsymbol{x}(k-1), \boldsymbol{u}(k-1), t_{k-1})\right] \tag{B.44}$$

implies that the old value

$$\dot{\boldsymbol{x}}(k-1) = \boldsymbol{f}(\boldsymbol{x}(k-1), \boldsymbol{u}(k-1), t_{k-1}) \tag{B.45}$$

must be stored. The global truncation error for this method is of order $O(h^2)$. The advantage with this method compared to Euler integration is seen from Figure B.1.

B.2.3 Runge–Kutta Second-Order Method

Heun's integration method or Runge–Kutta's second-order method (RK-2) is implemented as

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k), t_k) \\ \boldsymbol{k}_2 &= \boldsymbol{f}\left(\boldsymbol{x}(k) + h\boldsymbol{k}_1, \boldsymbol{u}(k), t_k + h\right) \\ \boldsymbol{x}(k+1) &= \boldsymbol{x}(k) + \tfrac{h}{2}(\boldsymbol{k}_1 + \boldsymbol{k}_2) \end{aligned} \tag{B.46}$$

The global truncation error for Heun's method is of order $O(h^2)$.

B.2.4 Runge–Kutta Fourth-Order Method

An extension of Heun's integration method to the fourth order (RK-4) is

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k), t_k) \\ \boldsymbol{k}_2 &= h\boldsymbol{f}(\boldsymbol{x}(k) + \boldsymbol{k}_1/2, \boldsymbol{u}(k), t_k + h/2) \\ \boldsymbol{k}_3 &= h\boldsymbol{f}(\boldsymbol{x}(k) + \boldsymbol{k}_2/2, \boldsymbol{u}(k), t_k + h/2) \\ \boldsymbol{k}_4 &= h\boldsymbol{f}(\boldsymbol{x}(k) + \boldsymbol{k}_3/2, \boldsymbol{u}(k), t_k + h) \\ \boldsymbol{x}(k+1) &= \boldsymbol{x}(k) + \frac{1}{6}(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4) \end{aligned} \tag{B.47}$$

The global truncation error for the RK-4 method is of order $O(h^4)$.

B.3 Numerical Differentiation

Numerical differentiation is usually sensitive to noisy measurements. Nevertheless, a reasonable estimate $\dot{\eta}_f$ of the time derivative $\dot{\eta}$ of a signal η can be obtained by using a *filtered differentiation*. The simplest filter is obtained by the first-order low-pass structure

$$\dot{\eta}_f(s) = \frac{Ts}{Ts+1}\,\eta(s) \tag{B.48}$$

corresponding to the continuous-time system

$$\dot{x} = ax + bu \tag{B.49}$$

$$y = cx + du \tag{B.50}$$

with $u = \eta$, $y = \dot{\eta}_f$, $a = b = -1/T$ and $c = d = 1$. Using the results from Example B.1, the following discrete-time filter equations can be used to differentiate a time-varying signal:

$$x(k+1) = \exp(-h/T)x(k) + [\exp(-h/T) - 1]u(k) \tag{B.51}$$

$$y(k) = x(k) + u(k) \tag{B.52}$$