Integrating from t to $t+h$ and multiplying by $\exp(-\gamma t)$, we arrive at the solution

$$\boldsymbol{p}(t+h) = e^{-\gamma h}\boldsymbol{p}(t) + \sqrt{\mathrm{k_B T}(1-e^{-2\gamma h})}\boldsymbol{M}^{1/2}\mathbf{Z}, \tag{7.3}$$

where $\mathbf{Z}$ is a vector of independent, standard normal random variables. Alternatively, we may write this as a discrete iteration in terms of a random map:

$$\boldsymbol{p}_{n+1} = e^{-\gamma h}\boldsymbol{p}_n + \sqrt{\mathrm{k_B T}(1-e^{-2\gamma h})}\boldsymbol{M}^{1/2}\mathbf{R}_n,$$

with normally distributed $\mathbf{R}$. For later purposes, we will think of this update as acting on both $\boldsymbol{q}$ and $\boldsymbol{p}$, leaving $\boldsymbol{q}$ unchanged, i.e.

$$(\boldsymbol{q}_{n+1}, \boldsymbol{p}_{n+1}) = (\boldsymbol{q}_n, e^{-\gamma h}\boldsymbol{p}_n + \sqrt{\mathrm{k_B T}(1-e^{-2\gamma h})}\boldsymbol{M}^{1/2}\mathbf{R}_n).$$

The Fokker-Planck operator for the OU system (7.2) can be computed directly as

$$\mathcal{L}_{\mathrm{O}}^{\dagger}\phi(\boldsymbol{q},\boldsymbol{p}) = \gamma\nabla_p\cdot(\boldsymbol{p}\phi) + \gamma\mathrm{k_B T}\nabla_p\cdot\boldsymbol{M}\nabla_p\phi.$$

The Ornstein-Uhlenbeck process models the momentum of a Brownian particle, and in deriving it we thought of γ as being associated to the collisional frequency with the particles of the bath. Alternatively, if we ignore the physical origin of this parameter, we can think of γ as an arbitrary coefficient; the sampling of the invariant measure does not depend on this choice. Thus we have a family of *sampling iterations* for the Gaussian distribution, defined for different choices of γ.

7.3.1 Splitting Methods for Langevin Dynamics

We will now consider a family of iterations for molecular systems which are constructed from simple building blocks based on the splitting of an SDE, specifically Langevin dynamics:

$$\begin{aligned} \mathrm{d}\boldsymbol{q} &= \boldsymbol{M}^{-1}\boldsymbol{p}\mathrm{d}t, \\ \mathrm{d}\boldsymbol{p} &= -\nabla U(\boldsymbol{q})\mathrm{d}t - \gamma\boldsymbol{p}\mathrm{d}t + \sqrt{2\gamma\mathrm{k_B T}}\boldsymbol{M}^{1/2}\mathrm{d}\mathbf{W}. \end{aligned} \tag{7.4}$$

Just as in the deterministic setting (for Newton's equations of motion) we construct such splitting methods by way of an additive decomposition of the vector field, where the differential equations corresponding to any individual piece can be solved exactly. Methods are built from a sequence of updates corresponding to an exact solve of each piece. By "exact solve" we here mean the construction of a method of exactly sampling the distribution generated by the corresponding component. The

choice of splitting strategy (given by the definition of the splitting pieces) along with a prescribed sequence of updates defines a splitting method.

There are a great number of plausible splitting strategies for Langevin dynamics. For example we may split the equations into purely deterministic and stochastic parts, break the stochastic terms into dissipative and random components, or choose to decompose the vector field into N_c one-dimensional problems that can be solved to high-accuracy. For some potential energy functions, it may be beneficial to decompose the potential into many terms $U(\boldsymbol{q}) = U_1(\boldsymbol{q}) + U_2(\boldsymbol{q}) + \ldots$, and use each term in a separate splitting piece in order to give a scheme improved stability or computational properties. In any case, it is not obvious *a priori* which splitting strategies will have more favorable sampling properties.

As propagating the Ornstein-Uhlenbeck process exactly will preserve the canonical distribution ρ_β, it is reasonable to assume that this is a good choice for one of the building blocks in a general splitting scheme. As the exact solution of the OU process is defined for any value of $\gamma \geq 0$, we may hope that our discretization scheme will be tolerant for extremely large frictions, although there is no guarantee that the resulting discretization remains consistent in the limiting case.

Other than the Ornstein-Uhlenbeck process, the remaining piece of the splitting corresponds to Newtonian constant-energy (microcanonical) Hamiltonian dynamics. Since Hamiltonian dynamics leaves invariant any function of the energy, its corresponding Fokker-Planck operator (in this case the Liouvillian, $\mathcal{L}_H^\dagger = -\mathcal{L}_H$) will preserve distributions that are functions of the Hamiltonian H. This implies in particular that it preserves ρ_β, which is proportional to $\exp(-\beta H)$. Thus the forward propagator associated to the Hamiltonian system automatically preserves the Gibbs distribution, and we have

$$e^{t\mathcal{L}_H^\dagger}\rho_\beta = \rho_\beta.$$

Formally, we could combine the flow map $\mathscr{F}_h^{\mathrm{H}}$ of Hamiltonian dynamics with an exact solve of the OU process to create a sampling iteration which preserves ρ_β, as we also have

$$e^{t\mathcal{L}_O^\dagger}\rho_\beta = \rho_\beta.$$

To make this practical in the general case, where the Hamiltonian system is not integrable (and we cannot explicitly compute its flow map), we need to be able to compute (or approximate) the maps involved.

This suggests to further split the Hamiltonian system, as we have done in deriving symplectic integrators in Chaps. 2 and 3.

We therefore consider dividing the Langevin system (7.4) into parts

$$\mathrm{d}\begin{bmatrix}\boldsymbol{q}\\ \boldsymbol{p}\end{bmatrix} = \underbrace{\begin{bmatrix}\boldsymbol{M}^{-1}\boldsymbol{p}\\ \mathbf{0}\end{bmatrix}\mathrm{d}t}_{\mathrm{A}} + \underbrace{\begin{bmatrix}\mathbf{0}\\ -\nabla U(\boldsymbol{q})\end{bmatrix}\mathrm{d}t}_{\mathrm{B}} + \underbrace{\begin{bmatrix}\mathbf{0}\\ -\gamma\boldsymbol{p}\mathrm{d}t + \sigma\boldsymbol{M}^{1/2}\mathrm{d}\mathbf{W}\end{bmatrix}}_{\mathrm{O}}, \tag{7.5}$$

which we label A, B and O, where each of the three parts may be solved "exactly", with individual updates given as

$$\begin{aligned}\mathscr{U}_h^{\mathrm{A}}(\boldsymbol{q},\boldsymbol{p}) &= (\boldsymbol{q} + h\boldsymbol{M}^{-1}\boldsymbol{p}, \boldsymbol{p}),\\ \mathscr{U}_h^{\mathrm{B}}(\boldsymbol{q},\boldsymbol{p}) &= (\boldsymbol{q}, \boldsymbol{p} - h\nabla U(\boldsymbol{q})),\\ \mathscr{U}_h^{\mathrm{O}}(\boldsymbol{q},\boldsymbol{p}) &= (\boldsymbol{q}, e^{-\gamma h}\boldsymbol{p} + \sqrt{\mathrm{k_B T}\,(1 - e^{-2\gamma h})}\boldsymbol{M}^{1/2}\mathbf{R}),\end{aligned}$$

where $\mathbf{R}$ is a vector of i.i.d. normal random numbers. For the deterministic pieces A and B, the updates correspond to their flow maps, while the update for the O piece corresponds to the solution derived in (7.3). The relevant forward propagators for each piece of the Hamiltonian part are computed through the adjoint of the Lie derivative as

$$\mathcal{L}_{\mathrm{A}}^{\dagger} = -\boldsymbol{M}^{-1}\boldsymbol{p}\cdot\nabla_q, \qquad \mathcal{L}_{\mathrm{B}}^{\dagger} = \nabla U(\boldsymbol{q})\cdot\nabla_p.$$

Using this splitting strategy, we can define a family of schemes by the desired sequence of updates, codified as a string using the alphabet "ABO". For example, the method denoted $[\![\mathrm{ABO}]\!]$ (acting from left to right) has a resulting update

$$\mathscr{U}_h^{[\![\mathrm{ABO}]\!]} = \mathscr{U}_h^{\mathrm{O}} \circ \mathscr{U}_h^{\mathrm{B}} \circ \mathscr{U}_h^{\mathrm{A}}.$$

This corresponds to using the adjoint symplectic Euler scheme to solve the Newtonian part of the Langevin dynamics SDE, followed by an exact OU solve. An alternative is to use velocity Verlet for the Hamiltonian part, resulting in a scheme denoted $[\![\mathrm{BABO}]\!]$ with associated map

$$\mathscr{U}_h^{[\![\mathrm{BABO}]\!]} = \mathscr{U}_h^{\mathrm{O}} \circ \mathscr{U}_{h/2}^{\mathrm{B}} \circ \mathscr{U}_h^{\mathrm{A}} \circ \mathscr{U}_{h/2}^{\mathrm{B}}.$$

Note that the presence of two B characters in the string results in halving the corresponding piece's time step length. As we need the method to be consistent, we require that the each of the A, B and O pieces are propagated in time for a total time of h. As a result we use the convention that if a letter appears k times in a method's string, then each individual update piece uses a timestep h/k. For example, the scheme denoted $[\![\mathrm{OABOAOBAO}]\!]$ has an update

$$\mathscr{U}_h^{[\![\mathrm{OABOAOBAO}]\!]} = \mathscr{U}_{h/4}^{\mathrm{O}} \circ \mathscr{U}_{h/3}^{\mathrm{A}} \circ \mathscr{U}_{h/2}^{\mathrm{B}} \circ \mathscr{U}_{h/4}^{\mathrm{O}} \circ \mathscr{U}_{h/3}^{\mathrm{A}} \circ \mathscr{U}_{h/4}^{\mathrm{O}} \circ \mathscr{U}_{h/2}^{\mathrm{B}} \circ \mathscr{U}_{h/3}^{\mathrm{A}} \circ \mathscr{U}_{h/4}^{\mathrm{O}},$$

where the random numbers used in each O update are independent. This scheme will require a total of $4\mathrm{N}_c$ normal random numbers and two force evaluations per step (the force must be recomputed at the first B step following an A step; the number of force evaluations required is not directly related to the length of the string).

A class of methods like those considered above were discussed in [46] and given the general title of "Geometric" Langevin Algorithm (GLA). The idea of a GLA

method is to treat the Newtonian part of Langevin dynamics by a symplectic method, and follow this by an Ornstein-Uhlenbeck exact solve; this is just what is done in the schemes denoted ⟦BABO⟧ and ⟦ABAO⟧. Thus we could give these two methods the alternate (and more descriptive) names Geometric Langevin (Velocity-Verlet) and Geometric Langevin (Position-Verlet) respectively.

Building on these schemes, the OU solves can instead be placed in the center of the step, or we may choose to have more than one OU solve distributed through the iteration. As it happens, some of these methods are of particular interest. We refer to the method obtained by inserting the OU solve in the middle of the Verlet method either by the strings ⟦ABOBA⟧ (Position-Verlet version) and ⟦BAOAB⟧ (Velocity-Verlet version) or by the more pronounceable names Symmetric Langevin Position-Verlet or Symmetric Langevin Velocity-Verlet, respectively.

For reference and to remove any ambiguity, we write out some of these algorithms below as single timestep maps (sample implementations in MATLAB are available at http://www.MolecularDynamics.info).

⟦ABAO⟧

$$\begin{aligned}
\boldsymbol{q}_{n+1/2} &= \boldsymbol{q}_n + (h/2)\boldsymbol{M}^{-1}\boldsymbol{p}_n,\\
\boldsymbol{p}_{n+1/2} &= \boldsymbol{p}_n - h\nabla U(\boldsymbol{q}_{n+1/2}),\\
\boldsymbol{q}_{n+1} &= \boldsymbol{q}_{n+1/2} + (h/2)\boldsymbol{M}^{-1}\boldsymbol{p}_{n+1/2},\\
\boldsymbol{p}_{n+1} &= e^{-h\gamma}\boldsymbol{p}_{n+1/2} + \zeta_2\boldsymbol{M}^{1/2}\mathbf{R}_n,
\end{aligned}$$

⟦BABO⟧

$$\begin{aligned}
\boldsymbol{p}_{n+1/2} &= \boldsymbol{p}_n - (h/2)\nabla U(\boldsymbol{q}_n),\\
\boldsymbol{q}_{n+1} &= \boldsymbol{q}_n + h\boldsymbol{M}^{-1}\boldsymbol{p}_{n+1/2},\\
\hat{\boldsymbol{p}}_{n+1/2} &= \boldsymbol{p}_{n+1/2} - (h/2)\nabla U(\boldsymbol{q}_{n+1}),\\
\boldsymbol{p}_{n+1} &= e^{-h\gamma}\hat{\boldsymbol{p}}_{n+1/2} + \zeta_2\boldsymbol{M}^{1/2}\mathbf{R}_n,
\end{aligned}$$

⟦ABOBA⟧

$$\begin{aligned}
\boldsymbol{q}_{n+1/2} &= \boldsymbol{q}_n + (h/2)\boldsymbol{M}^{-1}\boldsymbol{p}_n,\\
\boldsymbol{p}_{n+1/2} &= \boldsymbol{p}_n - (h/2)\nabla U(\boldsymbol{q}_{n+1/2}),\\
\hat{\boldsymbol{p}}_{n+1/2} &= e^{-h\gamma}\boldsymbol{p}_{n+1/2} + \zeta_2\boldsymbol{M}^{1/2}\mathbf{R}_n,\\
\boldsymbol{p}_{n+1} &= \hat{\boldsymbol{p}}_{n+1/2} - (h/2)\nabla U(\boldsymbol{q}_{n+1/2}),\\
\boldsymbol{q}_{n+1} &= \boldsymbol{q}_{n+1/2} + (h/2)\boldsymbol{M}^{-1}\boldsymbol{p}_{n+1},
\end{aligned}$$

⟦BAOAB⟧

$$\begin{aligned}
\boldsymbol{p}_{n+1/2} &= \boldsymbol{p}_n - (h/2)\nabla U(\boldsymbol{q}_n),\\
\boldsymbol{q}_{n+1/2} &= \boldsymbol{q}_n + (h/2)\boldsymbol{M}^{-1}\boldsymbol{p}_{n+1/2},\\
\hat{\boldsymbol{p}}_{n+1/2} &= e^{-h\gamma}\boldsymbol{p}_{n+1/2} + \zeta_2\boldsymbol{M}^{1/2}\mathbf{R}_n,\\
\boldsymbol{q}_{n+1} &= \boldsymbol{q}_{n+1/2} + (h/2)\boldsymbol{M}^{-1}\hat{\boldsymbol{p}}_{n+1/2},\\
\boldsymbol{p}_{n+1} &= \hat{\boldsymbol{p}}_{n+1/2} - (h/2)\nabla U(\boldsymbol{q}_{n+1}),
\end{aligned}$$

⟦OABAO⟧

$$\begin{aligned}
\boldsymbol{p}_{n+1/2} &= e^{-h\gamma/2}\boldsymbol{p}_n + \zeta_1\boldsymbol{M}^{1/2}\mathbf{R}_n,\\
\boldsymbol{q}_{n+1/2} &= \boldsymbol{q}_n + (h/2)\boldsymbol{M}^{-1}\boldsymbol{p}_{n+1/2},\\
\hat{\boldsymbol{p}}_{n+1/2} &= \boldsymbol{p}_{n+1/2} - h\nabla U(\boldsymbol{q}_{n+1/2}),\\
\boldsymbol{q}_{n+1} &= \boldsymbol{q}_{n+1/2} + (h/2)\boldsymbol{M}^{-1}\hat{\boldsymbol{p}}_{n+1/2},\\
\boldsymbol{p}_{n+1} &= e^{-h\gamma/2}\hat{\boldsymbol{p}}_{n+1/2} + \zeta_1\boldsymbol{M}^{1/2}\mathbf{R}_{n+1/2},
\end{aligned}$$

⟦OBABO⟧

$$\begin{aligned}
\boldsymbol{p}_{n+1/2} &= e^{-h\gamma/2}\boldsymbol{p}_n + \zeta_1\boldsymbol{M}^{1/2}\mathbf{R}_n,\\
\hat{\boldsymbol{p}}_{n+1/2} &= \boldsymbol{p}_{n+1/2} - (h/2)\nabla U(\boldsymbol{q}_n),\\
\boldsymbol{q}_{n+1} &= \boldsymbol{q}_n + h\boldsymbol{M}^{-1}\hat{\boldsymbol{p}}_{n+1/2},\\
\hat{\boldsymbol{p}}_{n+1} &= \hat{\boldsymbol{p}}_{n+1/2} - (h/2)\nabla U(\boldsymbol{q}_{n+1}),\\
\boldsymbol{p}_{n+1} &= e^{-h\gamma/2}\hat{\boldsymbol{p}}_{n+1} + \zeta_1\boldsymbol{M}^{1/2}\mathbf{R}_{n+1/2},
\end{aligned}$$

where $\zeta_j = \left[\mathrm{k_B T}\left(1 - e^{-j\gamma h}\right)\right]^{1/2}$, and $\mathbf{R}_n, \mathbf{R}_{n+1/2}$ are $\mathrm{N_d}$ vectors of standard independent $\mathcal{N}(0,1)$ random variables, resampled at each step. Note that, as in the Velocity-Verlet method, the two apparent evaluations of the force $\boldsymbol{F} = -\nabla U$ in the methods labelled ⟦BABO⟧, ⟦BAOAB⟧ and ⟦OBABO⟧ actually require only a single evaluation at each timestep, since the second evaluation at one step may be reused at the first stage of the following step. Thus all of the above methods require only one force evaluation per timestep, and hence we can think of them having exactly the same computational cost as the evaluation of the force is the principle computational expense.

When iterated $n+1$ times, we can see the scheme $[\![\text{OABA}]\!]$ produces a mapping of the form

$$\left(\mathscr{U}_h^{[\![\text{OABA}]\!]}\right)^{n+1} = \mathscr{U}_{h/2}^{\text{A}} \circ \mathscr{U}_{h/2}^{\text{B}} \circ \left(\mathscr{U}_h^{[\![\text{BAOAB}]\!]}\right)^{n} \circ \mathscr{U}_{h/2}^{\text{B}} \circ \mathscr{U}_{h/2}^{\text{A}} \circ \mathscr{U}_h^{\text{O}},$$

therefore through a coordinate transformation the $[\![\text{OABA}]\!]$ method can be morphed into the $[\![\text{BAOAB}]\!]$ scheme. To put this another way, if we compute an iteration using $[\![\text{OABA}]\!]$ and simply evaluate it at different points within the step, we can recover the same results as the $[\![\text{BAOAB}]\!]$ method would provide. Another term to describe this relationship is to say that one method is *conjugate* to the other.

7.3.2 Other Langevin Integrators

We may use our alphabet of A, B and O splitting pieces to create a wide variety of methods other than those given in the previous section (some requiring multiple force evaluations). However, it is not obvious that there is any gain in increasing the complexity of the splitting scheme (in terms of the length of its codifying string). Some popular methods can be rewritten in terms of this splitting, for example the scheme considered in [62] is equivalent to $[\![\text{OBABO}]\!]$. Examples of methods based on this type of splitting were also proposed by De Fabritiis et al [96] and Thalmann and Farago [363], in the context of dissipative particle dynamics (see Chapter 8).

Other Langevin dynamics splitting schemes in common use utilize a different additive decomposition of the SDE vector field. For example, the Stochastic Position Verlet (SPV) method of [265] relies on the splitting strategy

$$\mathrm{d}\begin{bmatrix}\boldsymbol{q}\\ \boldsymbol{p}\end{bmatrix} = \begin{bmatrix}\boldsymbol{M}^{-1}\boldsymbol{p}\\ \mathbf{0}\end{bmatrix}\mathrm{d}t + \begin{bmatrix}\mathbf{0}\\ -\nabla U(\boldsymbol{q})\mathrm{d}t - \gamma\boldsymbol{p}\mathrm{d}t + \sigma\boldsymbol{M}^{1/2}\mathrm{d}\mathbf{W}\end{bmatrix}.$$

To solve the splitting piece including the stochastic $\mathrm{d}\mathbf{W}$ term, we note that $\boldsymbol{q}$ does not change through its integration, and hence we may treat $\boldsymbol{F} = -\nabla U$ as constant during this part of the step. The exact solution is hence given through the usual Ornstein-Uhlenbeck formula, where the scheme itself derives from a symmetric integration of the two parts:

Stochastic Position Verlet (SPV)

$$\begin{aligned}\boldsymbol{q}_{n+1/2} &= \boldsymbol{q}_n + (h/2)\boldsymbol{M}^{-1}\boldsymbol{p}_n\\ \boldsymbol{p}_{n+1} &= e^{-h\gamma}\boldsymbol{p}_n - \eta\nabla U(\boldsymbol{q}_n) + \zeta\boldsymbol{M}^{1/2}\mathbf{R}_n\\ \boldsymbol{q}_{n+1} &= \boldsymbol{q}_{n+1/2} + (h/2)\boldsymbol{M}^{-1}\boldsymbol{p}_{n+1}\end{aligned}$$

where $\eta = (1-e^{-h\gamma})/\gamma$, $\zeta = \left[k_{\text{B}}\text{T}\left(1-e^{-2\gamma h}\right)\right]^{1/2}$. It is important to note that in the limit of infinite friction, we have $\eta \to 0$ independent of the timestep h. This suggests that this splitting strategy is unsuitable for large choices of the parameter γ as the

sampling will become inconsistent: the force will be dominated by the stochastic term, leaving the dynamics to evolve as a random walk.

Similar behavior is exhibited using the splitting strategy in (7.5), when interposing the *kick* term B between two O steps, such as in the schemes ⟦AOBOA⟧ or ⟦BOAOB⟧. By avoiding "OBO" updates, we can derive schemes that are consistent even at infinite friction (the ⟦BAOAB⟧ and ⟦ABOBA⟧ schemes are examples of such methods).

The Brünger-Brooks-Karplus (BBK) Langevin integrator [55] is one of the most popular and widely-implemented schemes.

Brünger-Brooks-Karplus (BBK)

$$\begin{aligned}
\boldsymbol{p}_{n+1/2} &= (1 - h\gamma/2)\boldsymbol{p}_n - (h/2)\nabla U(\boldsymbol{q}_n) + \tfrac{1}{2}\sqrt{2k_{\mathrm{B}}\mathrm{T}h\gamma}\boldsymbol{M}^{1/2}\mathbf{R}_n \\
\boldsymbol{q}_{n+1} &= \boldsymbol{q}_n + h\boldsymbol{M}^{-1}\boldsymbol{p}_{n+1/2} \\
\boldsymbol{p}_{n+1} &= [\boldsymbol{p}_{n+1/2} - (h/2)\nabla U(\boldsymbol{q}_{n+1}) + \tfrac{1}{2}\sqrt{2k_{\mathrm{B}}\mathrm{T}h\gamma}\boldsymbol{M}^{1/2}\mathbf{R}'_n]/(1 + h\gamma/2)
\end{aligned}$$

Often $\mathbf{R}'$ is chosen such that $\mathbf{R}'_n = \mathbf{R}_{n+1}$, so the random vector would be reused at the start of the following timestep, however there are many variants of this scheme that incorporate the redraw of random numbers differently. In other variants of BBK we take $\mathbf{R}'_n = \mathbf{R}_n$, reusing the random number within the step. We may also consider independent $\mathbf{R}'_n$ and $\mathbf{R}_n$, however when using independent noise processes we must modify the algorithm as it will not sample canonically (see Exercise 1).

7.4 Error Results for Harmonic Problems

For general systems it can be very challenging to give a complete description of the weak error in averages computed from trajectories evolved through a Langevin dynamics discretization scheme. However, using a linear model problem significantly reduces the complexity, while using small dimensional test problems allows us to effectively annihilate sampling error in demonstrative numerical experiments.

As such, we return now to the harmonic oscillator, which as well as being the simplest molecular model is also one of the most relevant for molecular dynamics applications, as many issues of stability and timestep in molecular dynamics simulations arise due to harmonic potentials used to model covalent bonds (such as in crystalline solids and biomolecules). The canonical distributions of position and momentum are also of a simple form (Gaussians), making such oscillators particularly amenable to analysis.

Recall that the one-dimensional harmonic oscillator with spring constant Ω^2 has potential energy function $U(q) = \Omega^2 q^2/2$, with Hamiltonian

$$H(q,p) = \frac{p^2}{2m} + \frac{\Omega^2 q^2}{2}, \tag{7.6}$$

where $(q,p) \in \mathbb{R}^2$ represent a particle's position and momentum, and $m > 0$ is the particle's mass. In this setting, the canonical distribution ρ_β is the product of two