

464 A Test problems

465 **Fermi–Pasta–Ulam–Tsingou:** This dynamical system is a model for a chain of $2m + 1$ alternating
 466 stiff and soft springs connecting $2m$ mass points. The chain is fixed in both ends [8, 40]. With the
 467 coordinate transformation suggested in [8, Ch. I.5.I] we have coordinates $[q, p]^T \in \mathbb{R}^{4m}$ where
 468 $q_i, i = 1, \dots, m$ represents a scaled displacement of the i -th stiff spring and $q_{i+m}, i = 1, \dots, m$
 469 represents a scaled expansion of the i -th spring. q_i represents their velocities. Letting ω be the angular
 470 frequency of the stiff spring, in general the Hamiltonian is given by

$$H(q, p) = \frac{1}{2} \sum_{i=1}^m (p_i^2 + p_{i+m}^2) + \frac{\omega^2}{2} \sum_{i=1}^m q_{i+m}^2 + \frac{1}{4} \left(\sum_{i=1}^{m-1} (q_{i+1} - q_{i+m+1} - q_i - q_{i+m})^4 + (q_1 - q_{m+1})^4 + (q_m + q_{2m})^4 \right)$$

471 We consider the most trivial case of $m = 1$ and letting $\omega = 2$, yielding the quartic, separable
 472 Hamiltonian by

$$H(q_1, q_2, p_1, p_2) = \frac{1}{2} (p_1^2 + p_2^2) + 2q_2^2 + \frac{1}{4} \left((q_1 - q_2)^4 + (q_1 + q_2)^4 \right).$$

473 **Double pendulum:** Let q_i and p_i denote the angle and angular momentum of pendulum $i = 1, 2$. The
 474 double pendulum system has a Hamiltonian that is not separable, where $y = [q_1, q_2, p_1, p_2]^T \in \mathbb{R}^4$
 475 and the Hamiltonian is given by

$$H(q_1, q_2, p_1, p_2) = \frac{\frac{1}{2}p_1^2 + p_2^2 - p_1 p_2 \cos(q_1 - q_2)}{1 + \sin^2(q_1 - q_2)} - 2\cos(q_1) - \cos(q_2).$$

476 **Hénon–Heiles:** This model was introduced for describing stellar motion inside the gravitational
 477 potential of a galaxy, as described in [8]. This Hamiltonian is separable. However, it is a canonical
 478 example of a chaotic system and its properties are discussed more in detail in [6]. The Hamiltonian is
 479 given by

$$H(q_1, q_2, p_1, p_2) = \frac{1}{2} (p_1^2 + p_2^2) + \frac{1}{2} (q_1^2 + q_2^2) + q_1^2 q_2 - \frac{1}{3} q_2^3.$$

480 B Additional numerical results

481 Here we present additional numerical experiments. In Figure 7, the flow error when learning from
 482 data without noise, could be found. The roll-out in time of the learned Hamiltonian for the FPUT and
 483 Hénon–Heiles problem is presented in Figure 8.

484 C More on numerical integration

485 C.1 Runge–Kutta methods

486 A general Runge–Kutta method for an autonomus system with s stages is a one-step numerical
 487 integrator given by

$$y_{n+1} = y_n + h \sum_{j=1}^s b_j k_j, \quad (16)$$

$$k_i = f\left(y_n + h \sum_{j=1}^s a_{ij} k_j\right), \quad i = 1, \dots, s.$$

488 A concrete method is determined by specifying the coefficient matrix $A \in \mathbb{R}^{s \times s}$ and the vector
 489 $b \in \mathbb{R}^s$, and there are conditions for symplecticity and order associated with these [41]. The
 490 conditions for order $p = 1$ require that the coefficient $c \in \mathbb{R}^s$ is determined by $c_i = \sum_{j=1}^s a_{ij}$. A

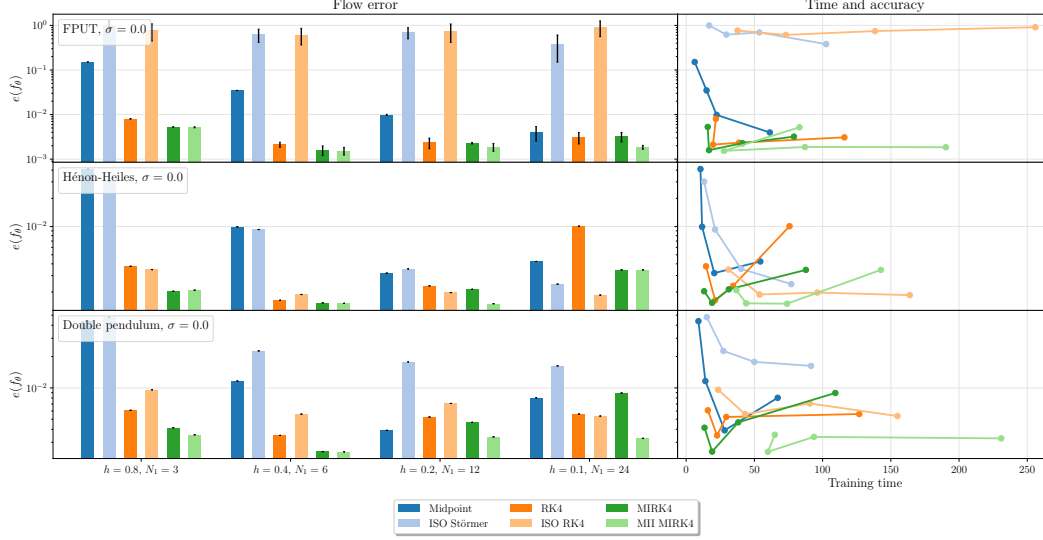


Figure 7: The flow error when learning vector fields using one-step methods directly (Midpoint, RK4 and MIRK4), ISO and multiple time-steps (ISO Störmer and ISO RK4) and MII (MII MIRK4). The error bars display the standard deviation after rerunning 5 experiments on data with $\sigma = 0$. The right subplot shows the computational time used in training against the flow error.

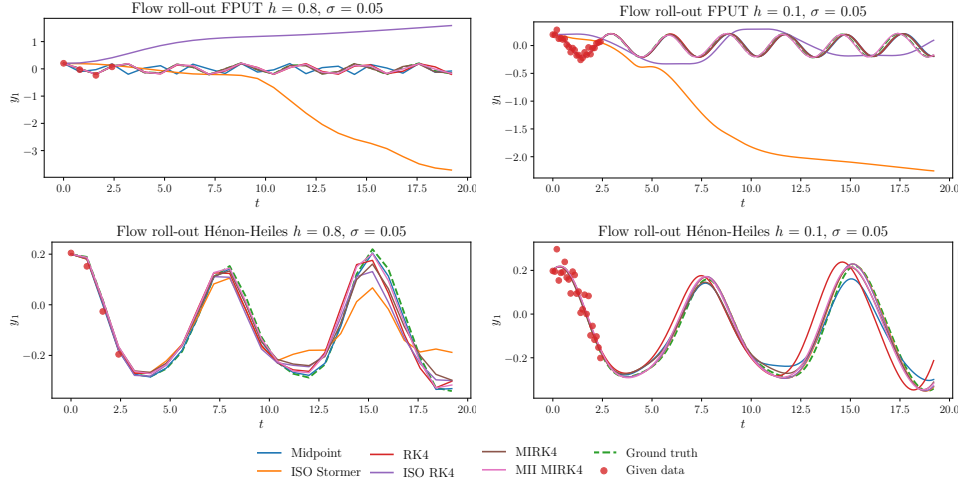


Figure 8: Roll-out in time obtained by integrating over the learned vector fields when training on data from the Fermi–Pasta–Ulam–Tsingou and Hénon–Heiles Hamiltonian.

491 method could be compactly represented by a *Butcher tableau* which structures the coefficients the
 492 following way

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array}$$

493 The two symplectic and symmetric Gauss-Legendre methods (found e.g. in [8]) with order $p = 4, 6$
 494 and denoted as GL4 and GL6 in Table I are presented in below:

495 C.2 Mono-Implicit Runge–Kutta methods

496 The MIRK methods are specified by a coefficient vector $b \in \mathbb{R}^s$, $v \in \mathbb{R}^s$ in addition to the strictly
 497 lower triangular matrix $D \in \mathbb{R}^{s \times s}$ and could be represented by the an extended Butcher tableau in

$\frac{1}{2} - \frac{\sqrt{3}}{6}$	$\frac{1}{4}$	$\frac{1}{4} - \frac{\sqrt{3}}{6}$	$\frac{1}{2} - \frac{\sqrt{15}}{10}$	$\frac{5}{36}$	$\frac{2}{9} - \frac{\sqrt{15}}{15}$	$\frac{5}{36} - \frac{\sqrt{15}}{30}$
$\frac{1}{2} + \frac{\sqrt{3}}{6}$	$\frac{1}{4} + \frac{\sqrt{3}}{6}$	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{5}{36} + \frac{\sqrt{15}}{24}$	$\frac{2}{9}$	$\frac{5}{36} - \frac{\sqrt{15}}{24}$
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2} + \frac{\sqrt{15}}{10}$	$\frac{2}{9} + \frac{\sqrt{15}}{15}$	$\frac{5}{36} + \frac{\sqrt{15}}{30}$	$\frac{5}{36}$
				$\frac{5}{18}$	$\frac{4}{9}$	$\frac{5}{18}$

Table 1: Properties of RK methods. *Symm.* is short for symmetric and *sympl.* for symplectic, and inv. for inverse.

Integration method	Name in figures	Order (p)	Stages (s)	Symm.	Sympl.	Inv. explicit	Explicit
Explicit Euler	E. Euler	1	1	no	no	yes	yes
Implicit Euler	I. Euler	1	1	no	no	yes	no
Runge–Kutta 4	RK4	4	4	no	no	yes	yes
Implicit midpoint	Midpoint	2	1	yes	yes	yes	no
MIRK3	MIRK3	3	2	no	no	yes	no
MIRK4	MIRK4	4	3	yes	no	yes	no
MIRK5	MIRK5	5	4	no	no	yes	no
MIRK6	MIRK6	6	5	yes	no	yes	no
Gauss Legendre 4	GL4	4	2	yes	yes	no	no
Gauss Legendre 6	GL6	6	4	yes	yes	no	no

the following manner

$$\begin{array}{c|c|c} c & v & D \\ \hline & & b^T \end{array}$$

In [22] it is proved that the maximum order of an s -stage MIRK method is $p = s + 1$ and several methods with stages $s \leq 5$ are presented. Below, we specify the MIRK methods used in the numerical experiments in addition to presenting their extended Butcher tableau in Figure 9.

- Midpoint: The symmetric and symplectic MIRK method where $(s, p) = (1, 2)$ is equivalent to the midpoint method.
- MIRK3: The method $(s, p) = (2, 3)$ found by choosing $c_1 = 1$ in [22].
- MIRK4: The method $(s, p) = (3, 4)$ with $x_{31} = \frac{1}{8}$ in [42] and is first presented in in [25, 43].
- MIRK5: The method $(s, p) = (4, 5)$ presented in [22] choosing $c_2 = 0$ and $c_3 = \frac{3}{2}$. It should be noted that as long as $c_3 > 1$ the method is A-stable, however the particular choice of $c_3 = \frac{3}{2}$ is arbitrary.
- MIRK6: The method $(s, p) = (5, 6)$ presented in [42], which is the $s = 5$ stage scheme in [22] choosing $c_3 = \frac{1}{2} - \frac{\sqrt{21}}{14}$. According to [42], this method is an improvement over earlier schemes on the same form which used $c_3 = \frac{1}{4}$.

C.3 Symmetric methods:

The exact flow of an ODE satisfies the following property known as (time) symmetry:

$$y(t_0) = \varphi_{h,f}^{-1}(y(t_0 + h)) = \varphi_{-h,f}(y(t_0 + h)),$$

where the superscript “ -1 ” denotes the inverse map. This is a desirable property also for the numerical approximation. A numerical integration method $\Phi_{h,f}$ is called symmetric if

$$\Phi_{h,f} = \Phi_{-h,f}^{-1}. \quad (17)$$

Symmetric numerical methods have the following properties [44]:

1. A symmetric integrator preserves the (time) symmetry of the exact flow.

[illegible]

Figure 9: Extended Butcher tableau of MIRK methods with stage and order $(s, p) = (2, 3), (3, 4), (4, 5), (5, 6)$.

519 2. The order p of a symmetric method is necessarily even.

3. Solutions of Hamiltonian systems satisfy the following reflection symmetry: if $(q(t), p(t))$ solves the Hamiltonian ODE, then $(q(-t), -p(-t))$ is also a solution, with $y(t) = [q(t), p(t)]^T$. Numerical solutions (q_n, p_n) obtained from a symmetric Runge–Kutta method satisfy the same reflection symmetry [7].

524 A Runge–Kutta method is symmetric if and only if

$$PA + AP - \mathbb{1}b^T = 0, \quad (18)$$

$$b = Pb, \quad (19)$$

where $\mathbb{1} := [1, \dots, 1]^T \in \mathbb{R}^s$ and $[p]_{ij} = \delta_{i, s+1-j}$ [44]. That is, P is the reflection of the identity matrix over the first axis. Inserting the definition of a MIRK method from (4), we get

$$\begin{aligned} PD + DP + (Pv + v - \mathbb{1})b^T &= 0 \\ b &= Pb. \end{aligned}$$

Symmetric MIRK methods of order $p = 2, 4, 6$ are presented in [45, 42] and specific examples are found in Figure 9

D Details on the inverse injection in MII

530 Assume we are deriving the MII following the example in Equation (10) using the implicit midpoint
531 method, where

$$y_{n+1} = y_n + hf\left(\frac{y_n + y_{n+1}}{2}\right) = y_n + h\Psi_{n,n+1}.$$

We thus find that the second term in (10), the composition of two steps starting in \tilde{y}_0 could be approximated by

$$\begin{aligned}
\hat{y}_2 &= \Phi_{h,f} \circ \Phi_{h,f}(\tilde{y}_0) \\
&= \Phi_{h,f}(\tilde{y}_0) + hf \left(\frac{\Phi_{h,f}(\tilde{y}_0) + \hat{y}_2}{2} \right) \\
&\approx \Phi_{h,f}(\tilde{y}_0) + hf \left(\frac{\tilde{y}_1 + \tilde{y}_2}{2} \right) \\
&= \tilde{y}_0 + hf \left(\frac{\tilde{y}_0 + \Phi_{h,f}(\tilde{y}_0)}{2} \right) + h\Psi_{1,2} \\
&\approx \tilde{y}_0 + hf \left(\frac{\tilde{y}_0 + \tilde{y}_1}{2} \right) + h\Psi_{1,2} \\
&= \tilde{y}_0 + h\Psi_{0,1} + h\Psi_{1,2}.
\end{aligned}$$

where the approximation \approx is obtained by the substitution $\tilde{y}_2 \rightarrow \hat{y}_2$ and $\tilde{y}_1 \rightarrow \Phi_{h,f}(\tilde{y}_0)$. The same procedure (repeatedly using the inverse injection) is generalized over longer trajectories and used to arrive at the MII method in Definition 5.1

E Details on neural network training

The experiments were performed on a Apple M1 Pro chip with double precision. The PyTorch L-BFGS [36] algorithm is run with the following parameters:

- History size: 120.
- Gradient tolerance: 10^{-9} .
- Termination tolerance on parameter changes: 10^{-9} .
- Line search: Strong Wolfe.

Both MII and ISO works better when f_θ has been pre-trained to be a reasonable approximation of the underlying vector field f . Thus, for both MII and ISO training is run 10 epochs on the one-step method before training additional 10 epochs with MII (MII MIRK4) and ISO (ISO Störmer and ISO RK4). The ISO procedure (searching for the optimal initial value \hat{y}_0) utilizes the L-BFGS optimization algorithm from the SciPy library [46] with gradient tolerance of 10^{-6} and the maximum number of iterations limited to 10.

F Proof of Theorem 4.4

Proof. As stated in Equation (5) Runge–Kutta method as given by Equation (16) is symplectic if and only if

$$b_i a_{ij} + b_j a_{ji} - b_i b_j = 0.$$

Inserting the particular form for the MIRK coefficients, $a_{ij} = d_{ij} + v_i b_j$, we get

$$\begin{aligned}
b_i(v_i b_j + d_{ij}) + b_j(v_j b_i + d_{ji}) - b_i b_j &= 0 \\
b_i d_{ij} + b_j d_{ji} + b_i b_j(v_j + v_i - 1) &= 0.
\end{aligned} \tag{20}$$

As D is strictly lower triangular either $d_{ji} = 0$ or $d_{ij} = 0$, which for Equation (20) implies that

$$\begin{cases} b_i d_{ij} + b_i b_j(v_j + v_i - 1) = 0 & \text{if } i \neq j \\ b_i^2(2v_i - 1) = 0 & \text{if } i = j \end{cases}$$

Requiring d_{ij} , b_i and v_i to satisfy the symplecticity condition yields the following restriction

$$\begin{cases} b_i d_{ij} + b_i b_j(v_j + v_i - 1) = 0 & \text{if } i \neq j, \\ b_i = 0 \text{ or } v_i = \frac{1}{2} & \text{if } i = j. \end{cases} \tag{21}$$

Without loss of generality, we assume that the m first entries of $b \in \mathbb{R}^s$ are zero. Enforcing the conditions of Equation (21) on $v \in \mathbb{R}^s$ we get for $1 \leq m \leq s$

$$b = [0, \dots, 0, b_{m+1}, \dots, b_s]^T,$$

$$v = [v_1, \dots, v_m, \frac{1}{2}, \dots, \frac{1}{2}]^T.$$

In total, this gives the following constraints for v, b, D :

	$b_j = 0$	$b_j \neq 0$
$b_i = 0$	$d_{ij} \in \mathbb{R}$ $v_i, v_j \in \mathbb{R}$	$d_{ij} \in \mathbb{R}$ $v_i, v_j \in \mathbb{R}$
$b_i \neq 0$	$d_{ij} = 0$ $v_i, v_j \in \mathbb{R}$	$d_{ij} = 0$ $v_i, v_j = \frac{1}{2}$

Which for the Runge–Kutta method $A = D + vb^T$ gives a (RK) Butcher tableau of the form

	0	0	...	0	$v_1 b_{m+1}$...	$v_1 b_s$
d_{21}	0	...	0				
d_{31}	d_{32}		0	\vdots			\vdots
\vdots			\ddots				
$d_{m,1}$...	$d_{m,m-1}$	0	$v_m b_{m+1}$...	$v_m b_s$	
	0	0	$\frac{1}{2} b_{m+1}$...	$\frac{1}{2} b_s$
\vdots				\vdots			\vdots
0	0	$\frac{1}{2} b_{m+1}$...	$\frac{1}{2} b_s$	
	0	0	b_{m+1}	...	b_s

Since the lower left submatrix is the zero matrix, this leaves the stages k_{m+1}, \dots, k_s unconnected to the first m stages. In addition, as $b_i = 0$ for $i = 1, \dots, m$, these stages are not included in the computation of the final integration step. The method is thus reducible to the lower right submatrix of A and b_{m+1}, \dots, b_s . The reduced method is thus in general given by the following stage-values

$$k_i = f(y_n + \frac{h}{2} \sum_j b_j k_j).$$

It is trivial to check that if $\sum_i b_i = 1$ the method satisfies order conditions up to order $p = 2$, which could be found in [8] to be

$$\sum_i b_i = 1, \quad \text{and} \quad \sum_{i,j} b_i a_{ij} = \frac{1}{2}.$$

However, the method fails to satisfy the first of the two conditions required for order $p = 3$, since

$$\sum_{i,j,k} b_i a_{ij} a_{ik} = \frac{1}{4} \sum_{i,j,k} b_i b_j b_k = \frac{1}{4} \neq \frac{1}{3}.$$

Hence, the maximum order of a symplectic MIRK method is $p = 2$. □

As a remark, it should be noted that the $s = 1$ stage, symplectic MIRK method found by setting $b_1 = 1, v_1 = \frac{1}{2}$ and $d_{11} = 0$ is simply the midpoint method $y_{n+1} = y_n + h k_1$ with $k_1 = f(\frac{y_n + y_{n+1}}{2})$.

570 **G Proof of Theorem 5.2**

571 *Proof.* Let $s_i(y_n, y_{n+1}) := y_n + v_i(y_{n+1} - y_n)$ and \tilde{y}_n be noisy data (9). Observe that we can obtain
 572 the following approximation to the MIRK stages (4) by

$$\begin{aligned}
 k_i &= f\left(\tilde{y}_n + v_i(\tilde{y}_{n+1} - \tilde{y}_n) + h \sum_{j=1}^s d_{ij} k_j\right) \\
 &= f\left(s_i(y_n, y_{n+1}) + s_i(\delta_n, \delta_{n+1}) + \mathcal{O}(h)\right) \\
 &= f(s_i(y_n, y_{n+1})) + f'(s_i(y_n, y_{n+1}))s_i(\delta_n, \delta_{n+1}) + \mathcal{O}(\|s(\delta_n, \delta_{n+1})\|^2) + \mathcal{O}(h) \\
 &= f(y_n) + f'(y_n)s_i(\delta_n, \delta_{n+1}) + \mathcal{O}(\|s(\delta_n, \delta_{n+1})\|^2) + \mathcal{O}(h).
 \end{aligned} \tag{22}$$

573 Where in the final equality we expand $y_{n+1} = y_n + hf(y_n) + \mathcal{O}(h^2)$ to find

$$\begin{aligned}
 f(s_i(y_n, y_{n+1})) &= f(y_n + v_i(y_{n+1} - y_n)) \\
 &= f(y_n + hv_i f(y_n) + \mathcal{O}(h^2)) \\
 &= f(y_n) + \mathcal{O}(h).
 \end{aligned}$$

574 And similarly for $f'(s_i(y_n, y_{n+1}))$. In total, this means that the next MIRK-step could be approxi-
 575 mated by

$$\begin{aligned}
 y_{n+1} &= \tilde{y}_n + h \sum_{i=1}^s b_i k_i \\
 &= \tilde{y}_n + h \sum_{i=1}^s b_i \left(f(y_n) + f'(y_n)s_i(\delta_n, \delta_{n+1}) \right) + \mathcal{O}(h\|s(\delta_n, \delta_{n+1})\|^2) + \mathcal{O}(h^2).
 \end{aligned} \tag{23}$$

576 First note, that if x is a multivariate normally distributed random variable $x \sim \mathcal{N}(0, \Sigma)$ then for a
 577 matrix $G \in \mathbb{R}^{n \times n}$ the variance of the linear transformation is given by $\text{Var}[Gx] := \text{Cov}[Gx, Gx] =$
 578 $G\Sigma G^T$. Now, using the approximation in Equation (23), we find the variance of the optimization
 579 target $\mathcal{T}_{n+1}^{\text{OS}} = \tilde{y}_{n+1} - \Phi_{h,f}(\tilde{y}_n, \tilde{y}_{n+1})$ by

$$\begin{aligned}
 \text{Var}\left[\tilde{y}_{n+1} - \tilde{y}_n - h \sum_{i=1}^s b_i k_i\right] &\approx \text{Var}\left[\delta_{n+1} - \delta_n - h \sum_{i=1}^s b_i \left(f'(y_n)s_i(\delta_n, \delta_{n+1}) \right)\right] \\
 &= \text{Var}\left[\left(I - hb^T v f'(y_n)\right)\delta_{n+1} - \left(I + hb^T(\mathbb{1} - v)f'(y_n)\right)\delta_n\right] \\
 &= \sigma^2 \left(I - hb^T v f'(y_n) \right) \left(I - hb^T v f'(y_n) \right)^T \\
 &\quad + \sigma^2 \left(I + hb^T(\mathbb{1} - v)f'(y_n) \right) \left(I + hb^T(\mathbb{1} - v)f'(y_n) \right)^T \\
 &= \sigma^2 \left[2I + hb^T(\mathbb{1} - 2v)(f'(y_n) + f'(y_n)^T) \right. \\
 &\quad \left. + h^2 \underbrace{\left((b^T v)^2 + (b^T(\mathbb{1} - v))^2 \right) f'(y_n) f'(y_n)^T}_{:= Q^{\text{OS}}} \right] \\
 &= \sigma^2 \left[2I + hb^T(\mathbb{1} - 2v)(f'(y_n) + f'(y_n)^T) + h^2 Q^{\text{OS}} \right].
 \end{aligned}$$

580 Here, $\mathbb{1} := [1, \dots, 1]^T \in \mathbb{R}^s$. This is the variance estimate we wanted to find for MIRK methods
 581 used as one-step integration schemes. Similarly, considering a point computed by the mean inverse

integrator \bar{y}_n , we find, using the stage approximation by Equation (22) that

$$\begin{aligned}\bar{y}_n &= \frac{1}{N} \sum_{\substack{j=0 \\ j \neq n}}^N \tilde{y}_j + \frac{h}{N} \sum_{j=0}^{N-1} w_{n,j} \sum_{l=1}^s b_l k_l \\ &\approx \frac{1}{N} \sum_{\substack{j=0 \\ j \neq n}}^N \tilde{y}_j + \frac{h}{N} \sum_{j=0}^{N-1} w_{n,j} \sum_{l=1}^s b_l \left(f(y_j) + f'(y_j) s_l(\delta_j, \delta_{j+1}) \right)\end{aligned}$$

Where we note that $w_{n,j} := [W]_{nj}$, from Definition 5.1 of the MII. Let $\bar{y}_n := [\bar{Y}]_n$. Computing the variance of the optimization target $\mathcal{T}_i^{\text{MI}} = \tilde{y}_n - \bar{y}_n$ we find, by introducing \bar{P}_{nj} to simplify notation, that

$$\begin{aligned}\text{Var}[\tilde{y}_n - \bar{y}_n] &\approx \text{Var}\left[\delta_n - \frac{1}{N} \sum_{\substack{j=0 \\ j \neq n}}^N \delta_j - \frac{h}{N} \sum_{j=0}^{N-1} w_{n,j} f'(y_j) \left(b^T (\mathbb{1} - v) \delta_j + b^T v \delta_{j+1} \right)\right] \\ &= \text{Var}\left[\frac{1}{N} \sum_{\substack{j=0 \\ j \neq n}}^N \left(I + h f'(y_j) \underbrace{\left(\tilde{w}_{n,j} b^T (\mathbb{1} - v) + \tilde{w}_{n,j-1} b^T v \right)}_{:= \bar{P}_{nj}} \right) \delta_j \right] \\ &\quad + \text{Var}\left[\left(I - \frac{h}{N} \underbrace{f'(y_n) \left(\tilde{w}_{n,n} b^T (\mathbb{1} - v) + \tilde{w}_{n,n-1} b^T v \right)}_{:= \bar{P}_{nn}} \right) \delta_n \right] \\ &= \text{Var}\left[\frac{1}{N} \sum_{\substack{j=0 \\ j \neq n}}^N \left(I + h \bar{P}_{nj} \right) \delta_j \right] + \text{Var}\left[\left(I - \frac{h}{N} \bar{P}_{nn} \right) \delta_n \right] \\ &= \frac{\sigma^2}{N^2} \sum_{\substack{j=0 \\ j \neq n}}^N \left(I + h \bar{P}_{nj} \right) \left(I + h \bar{P}_{nj} \right)^T + \sigma^2 \left(I - \frac{h}{N} \bar{P}_{nn} \right) \left(I - \frac{h}{N} \bar{P}_{nn} \right)^T.\end{aligned}$$

In the second line, $\tilde{w}_{n,j}$ is introduced which is elements of a matrix $\tilde{W} = [0|w_1|w_2|\dots|w_N|0] \in \mathbb{R}^{N \times N+1}$, or in other words the matrix you obtain by padding W right and left with a column of zeros. Expanding the terms and introducing matrices P_{nj} and Q^{MI} we finally find

$$\begin{aligned}\text{Var}[\tilde{y}_n - \bar{y}_n] &\approx \frac{\sigma^2}{N} \left[(1+N)I + h \underbrace{(\bar{P}_{nn} + \bar{P}_{nn}^T)}_{= P_{nn}} + \frac{h}{N} \sum_{\substack{j=0 \\ j \neq n}}^s \underbrace{(\bar{P}_{nj} + \bar{P}_{nj}^T)}_{:= P_{nj}} + \frac{h^2}{N} \sum_{j=0}^s \underbrace{\bar{P}_{nj} \bar{P}_{nj}^T}_{:= Q^{\text{MI}}} \right] \\ &= \frac{\sigma^2}{N} \left[(1+N)I + h P_{nn} + \frac{h}{N} \sum_{\substack{j=0 \\ j \neq n}}^s P_{nj} + \frac{h^2}{N} Q^{\text{MI}} \right].\end{aligned}$$

Since for a symmetric matrix A we have that the spectral radii ρ (largest absolute value of eigenvalues) could be found by $\rho(A) = \|A\|_2$, we find for both variance approximations (covariance matrix is always symmetric) that

$$\begin{aligned}\rho\left(\text{Var}[\tilde{y}_n - \bar{y}_n]\right) &\approx \frac{\sigma^2}{N} \left\| (1+N)I + h P_{nn} + \frac{h}{N} \sum_{\substack{j=0 \\ j \neq n}}^s P_{nj} + \frac{h^2}{N} Q^{\text{MI}} \right\|_2 \\ \rho\left(\text{Var}[\tilde{y}_{n+1} - \Phi_{h,f}(\tilde{y}_n, \tilde{y}_{n+1})]\right) &\approx \sigma^2 \left\| 2I + h b^T (\mathbb{1} - 2v) (f'(y_n) + f'(y_n)^T) + h^2 Q^{\text{OS}} \right\|_2.\end{aligned}$$

Finally, we note that:

$$\begin{aligned}
Q^{\text{OS}} &:= \left((b^T v)^2 + (b^T (\mathbb{1} - v))^2 \right) f'(y_n) f'(y_n)^T \\
\bar{P}_{nj} &:= f'(y_j) \left(\tilde{w}_{n,j} b^T (\mathbb{1} - v) + \tilde{w}_{n,j-1} b^T v \right) \\
P_{nj} &:= \bar{P}_{nj} + \bar{P}_{nj}^T \\
Q^{\text{MII}} &:= \sum_{j=0}^s \bar{P}_{nj} \bar{P}_{nj}^T.
\end{aligned} \tag{24}$$

593

□

594 H Higher-order inverse-explicit invariant-preserving symmetric 595 non-partitioned integrators

596 We define invariant-preserving integrators as methods that preserve the Hamiltonian or other invariants
597 of the exact solution, either exactly up to machine precision or within a bound, like symplectic
598 methods. Although we argue in this paper that symplecticity is a less important property when
599 learning Hamiltonian systems from data than for integration of a known system, we do not mean to
600 suggest that invariant-preserving integrators may not be beneficial to some extent and have important
601 qualities in the inverse problem also. However, we urge anyone who seeks to use invariant-preserving
602 methods to also consider the order of the method and whether it is a symmetric inverse-explicit
603 method. Although the maximum order of a symplectic inverse-explicit *Runge–Kutta* method is two,
604 there exist higher-order inverse-explicit invariant-preserving integrators that are not Runge–Kutta
605 methods.

606 Note that *partitioned* Runge–Kutta (PRK) methods is an extension that does not belong to the class of
607 Runge–Kutta methods. This is important to clarify, since there exist PRK methods that are symmetric
608 and explicit for separable systems. This marks a distinction from non-partitioned RK methods:
609 these cannot be symmetric and explicit in general [8]. Several papers suggest using symplectic PRK
610 methods for learning Hamiltonian systems [10, 30, 29], but these methods, although symmetric, only
611 depend on one point to approximate the right hand side of each integration step, and thus do not
612 average out any noise.

613 H.1 Symplectic elementary differential Runge–Kutta methods

614 Chartier et al. showed in [47] that an integrator can be applied to a modified vector field in such a way
615 that it yields a higher order approximation of the original vector field while inheriting the geometric
616 properties of the given integrator. As an example, they present the fourth-order modified implicit
617 midpoint method

$$\frac{y_{n+1} - y_n}{h} = f(\bar{y}) + \frac{h}{12} \left(-Df(\bar{y})Df(\bar{y})f(\bar{y}) + \frac{1}{2}D^2f(\bar{y})f(\bar{y})f(\bar{y}) \right), \tag{25}$$

618 where $\bar{y} = (y_n + y_{n+1})/2$. This is an example of an elementary differential Runge–Kutta (EDRK)
619 method [33], which relies on the calculation of (multi-order) derivatives of the vector field f , denoted
620 here as $D^p f$ for order p . Automatic differentiation can be utilized also to get higher-order derivatives,
621 and we note that f , Df and $D^2 f$ each only have to be evaluated once for each training step, since
622 they are only evaluated at the one point \bar{y} . A sixth-order modification of the implicit midpoint method
623 is also presented in [47], but that requires the calculation of up to fourth-order derivatives and might
624 be considered prohibitively expensive.

625 H.2 Discrete gradient methods

626 Discrete gradient methods are a class of integrators that can preserve an invariant, e.g. the Hamiltonian,
627 exactly [32]. This is in contrast to symplectic methods, which only preserve a perturbation of the
628 invariant exactly and the exact invariant within some bound. We remark that no method can be both

629 symplectic and exactly invariant-preserving in general [48]. Discrete gradient methods are defined
 630 strictly for invariant-preserving ODEs, which can be written on the form

$$\dot{y} = S(y)\nabla H(y), \quad (26)$$

631 for some skew-symmetric matrix $S(y)$ [32]. Then a discrete gradient is a function $\bar{\nabla}H : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$
 632 satisfying

$$\bar{\nabla}H(u, v)^T(u - v) = H(u) - H(v),$$

633 a discrete analogue to the invariant-preserving property $\dot{H}(y) = \nabla H(y)^T \dot{y} = 0$ of (26). A corre-
 634 sponding discrete gradient method is then given by

$$\frac{y_{n+1} - y_n}{h} = \bar{S}(y_n, y_{n+1}, h)\bar{\nabla}H(y_n, y_{n+1}), \quad (27)$$

635 for some approximation $\bar{S}(y_n, y_{n+1}, h)$ of $S(y)$ such that $\bar{S}(y, y, 0) = S(y)$, where h is the step
 636 size in time. A discrete gradient can at most be a second-order approximation of the gradient, but
 637 appropriate choices of \bar{S} can yield inverse-explicit integrators of arbitrarily high order [16]. Matsubara
 638 et al. have developed a discrete version of the automatic differentiation algorithm that makes it possible
 639 to efficiently calculate a discrete gradient of neural network functions, and demonstrated its use in
 640 training of HNNs [15] and for detecting invariants [49]. A fourth-order discrete gradient method is
 641 suggested for training HNNs in [16], given a constant S in (26). This is the scheme (27) with

$$\bar{S}(y_n, \cdot, h) = S + \frac{8}{9}hSQ(y_n, z_2)S - \frac{1}{12}h^2SD^2H(z_1)SD^2H(z_1)S,$$

642 with $z_1 = y_n + \frac{1}{2}hf(y_n)$, $z_2 = y_n + \frac{3}{4}hf(z_1)$ and $Q(u, v) := \frac{1}{2}(D_2\bar{\nabla}H(u, v)^T - D_2\bar{\nabla}H(u, v))$,
 643 where $D_2\bar{\nabla}H$ denotes the derivative of $\bar{\nabla}H$ with respect to the second argument, and $D^2H := D\nabla H$
 644 is the Hessian of H . This is not symmetric, so we propose here instead the *fourth-order symmetric*
 645 *invariant-preserving scheme* obtained by

$$\begin{aligned} \bar{S}(y_n, y_{n+1}, h) = & S + \frac{h}{2}S(Q(y_n, \frac{1}{3}y_n + \frac{2}{3}y_{n+1}) - Q(y_{n+1}, \frac{2}{3}y_n + \frac{1}{3}y_{n+1}))S \\ & - \frac{1}{12}(h)^2SD^2H(\bar{y})SD^2H(\bar{y})S. \end{aligned}$$

646 H.3 Numerical comparison of fourth-order integrators

647 We test four different fourth-order integrators on solving an initial value problem of the double
 648 pendulum described in Appendix A. We compute an approximation of the error of the solution at each
 649 time by comparing to a solution obtained using RK4 with 10 times as many time steps. As seen in the
 650 left plot of Figure 10, the symmetric methods are clearly superior to the explicit RK4 method, when
 651 using the same step size. For integration, the advantage of RK4 is that it is more computationally
 652 efficient than the implicit methods, which facilitates taking smaller step sizes. However, as pointed
 653 out in Section 4, RK4 does not have this advantage over MIRK methods for the inverse problem.

654 Furthermore, although the higher-order MIRK methods we suggest to use in this paper are not
 655 symplectic and thus lack general energy preservation guarantees, we see from Figure 10 that they
 656 may still preserve the energy within a bound for specific problems. In fact, for the double pendulum
 657 problem considered here, the non-symplectic MIRK4 method preserves the energy slightly better
 658 than the symplectic MIMP4 scheme up to time $T = 500$. The invariant-preserving discrete gradient
 659 method preserves the Hamiltonian to machine precision.

660 I Computational cost

661 The fourth-order MIRK method from Table 9 (MIRK4) is between twice and thrice as expensive as
 662 the implicit midpoint method, depending on the training strategy. That is, if no batching is performed
 663 and f is evaluated at all points in the training set at each iteration of the optimization, then the number
 664 of function evaluations for a trajectory with n points is $n - 1$ for the implicit midpoint method and
 665 $2n - 1$ for MIRK4. However, if batching is done and function evaluations cannot generally be reused
 666 for successive points, the total number of function evaluations at each epoch may increase to $3n - 3$
 667 for MIRK4.

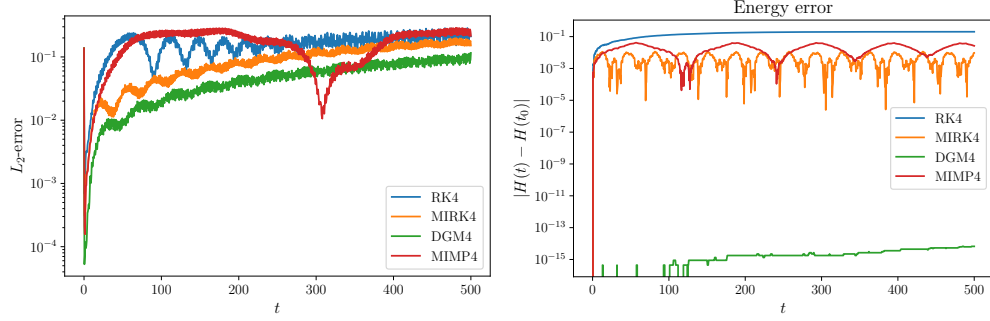


Figure 10: Global error (*left*) and energy error (*right*) of the solution of the double pendulum problem obtained using four different integrators. The initial condition is $y_0 = [0.1, 0.3, -0.4, 0.2]^T$, and the step size for all integrators is $h = \frac{1}{2}$.

668 In general, the cost of an s -stage MIRK method depends on both the training strategy and whether
669 the end points y and \hat{y} are two of the stages. If batching is not done and y and \hat{y} are two of the stages,
670 then computational cost at each epoch is $\mathcal{O}(m(n + (s - 2)(n - 1)))$, where m is the number of
671 trajectories of n points in each. The maximum cost with batching is the same as the cost if y and \hat{y}
672 are not two of the stages: $\mathcal{O}(ms(n - 1))$. This cost is equivalent to that of an explicit s -stage RK
673 method.