Rössler attractor

continuous deterministic chaos

Rössler system

$$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + ay \\ \dot{z} = b + z(x - c) \end{cases}$$

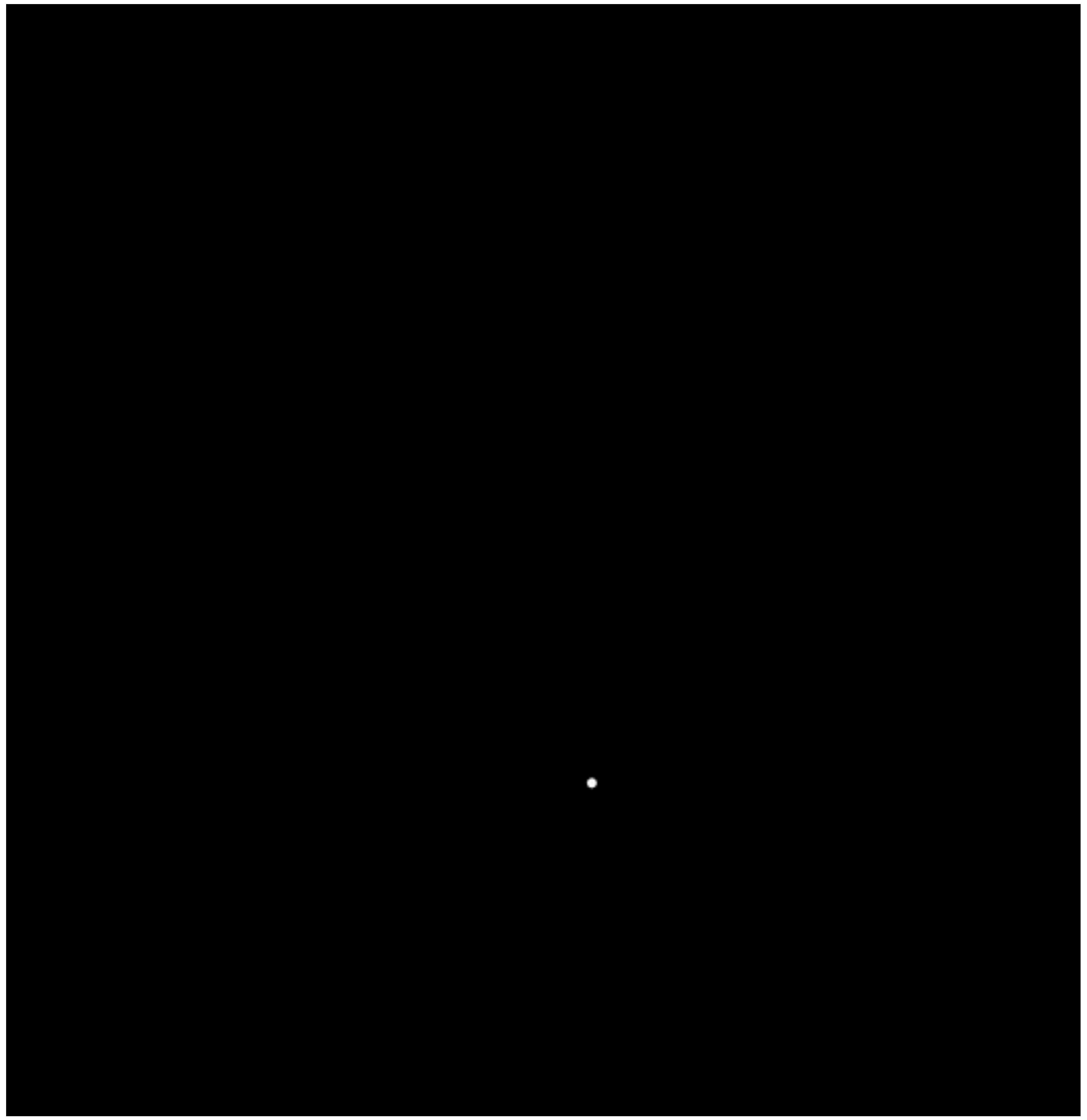
$\{x, y, z\}$ system state

$\{a, b, c\}$ parameters

$\{\dot{x}, \dot{y}, \dot{z}\} = \{0,0,0\}$ if $\{x, y, z\}$ equal to:

$$\left(\frac{c+\sqrt{c^2-4ab}}{2}, \frac{-c-\sqrt{c^2-4ab}}{2a}, \frac{c+\sqrt{c^2-4ab}}{2a}\right)$$

$$\left(\frac{c-\sqrt{c^2-4ab}}{2}, \frac{-c+\sqrt{c^2-4ab}}{2a}, \frac{c-\sqrt{c^2-4ab}}{2a}\right)$$



Rössler system

$$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + ay \\ \dot{z} = b + z(x - c) \end{cases}$$

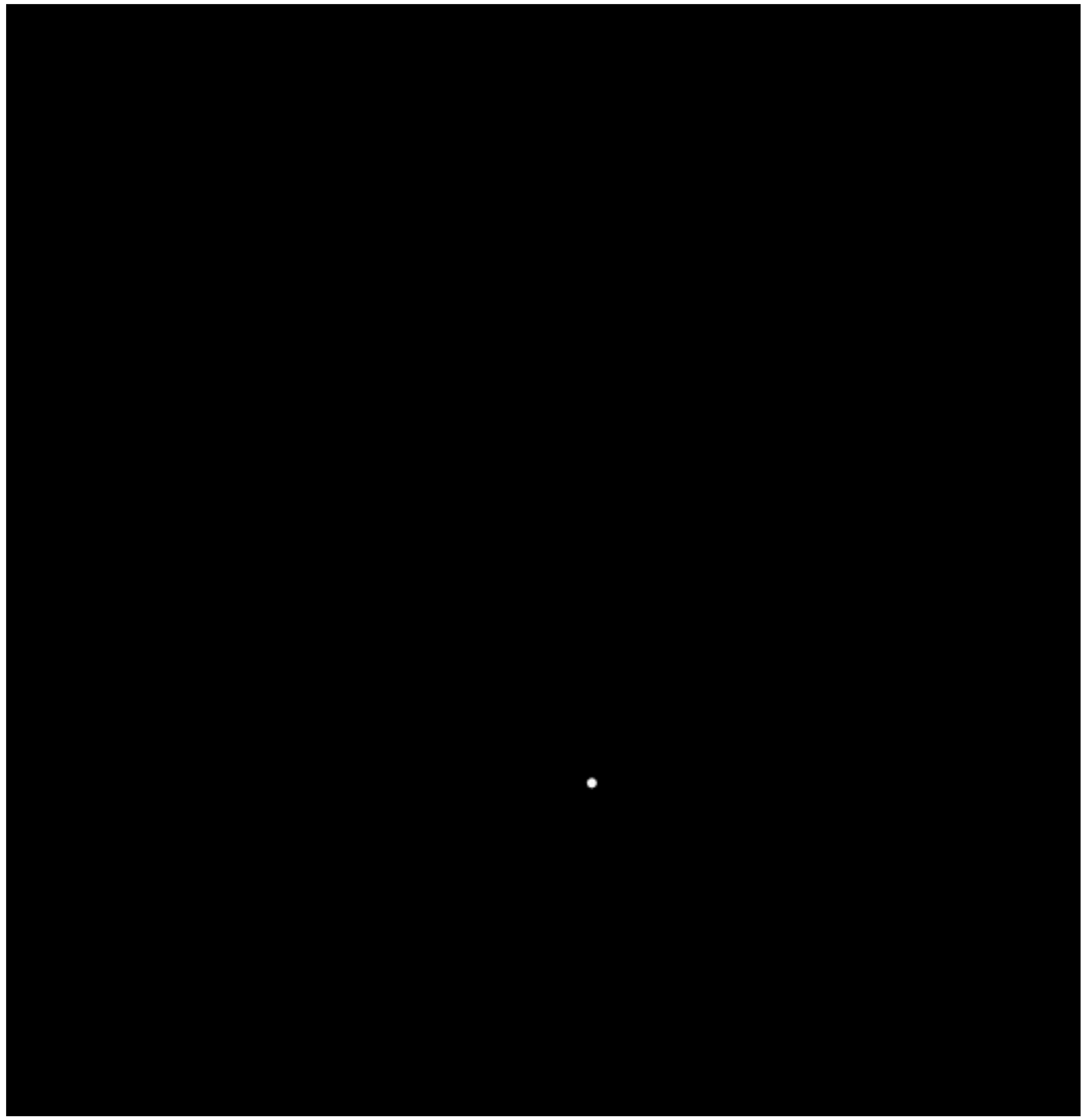
$\{x, y, z\}$ system state

$\{a, b, c\}$ parameters

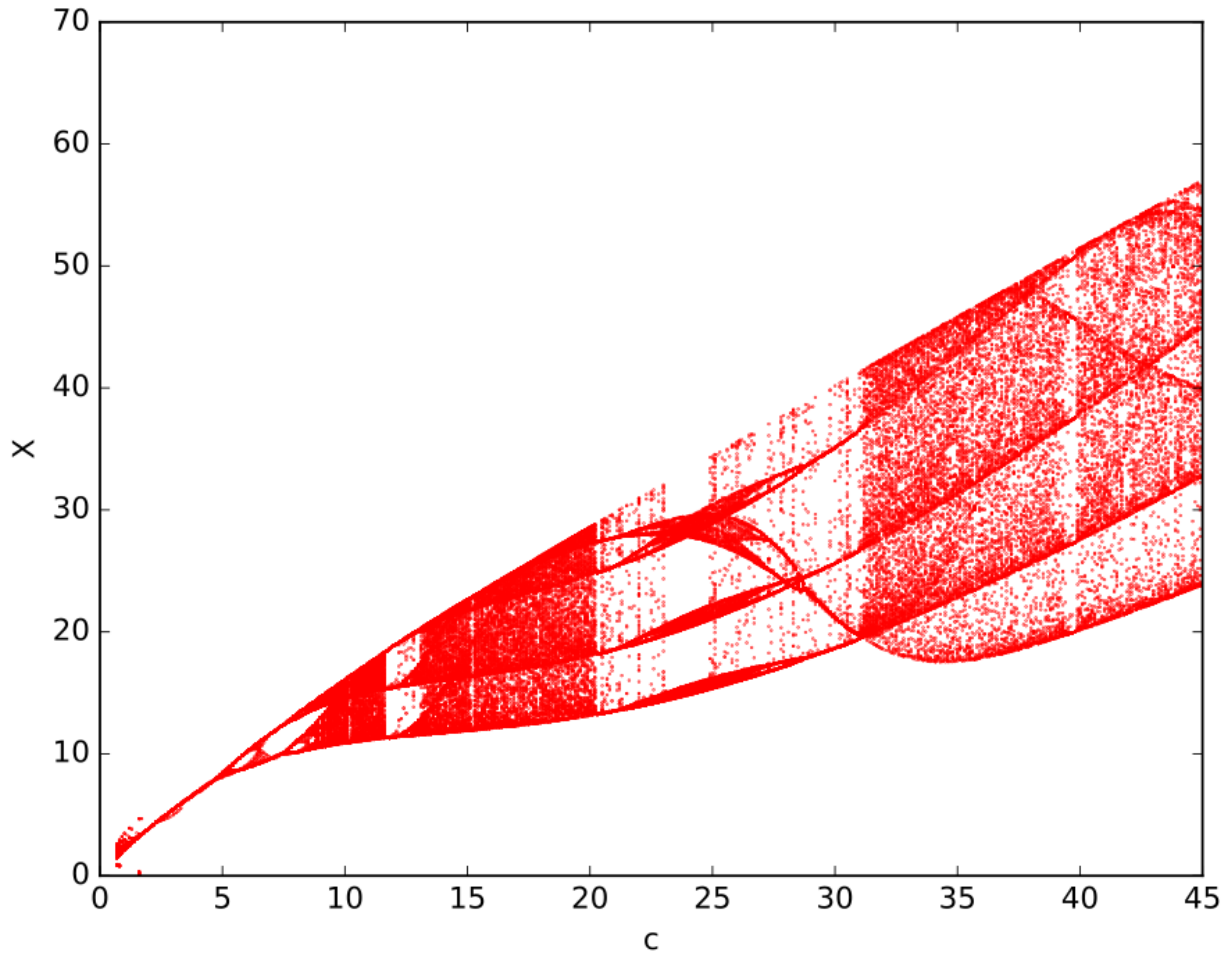
$\{\dot{x}, \dot{y}, \dot{z}\} = \{0,0,0\}$ if $\{x, y, z\}$ equal to:

$$\left(\frac{c+\sqrt{c^2-4ab}}{2}, \frac{-c-\sqrt{c^2-4ab}}{2a}, \frac{c+\sqrt{c^2-4ab}}{2a}\right)$$

$$\left(\frac{c-\sqrt{c^2-4ab}}{2}, \frac{-c+\sqrt{c^2-4ab}}{2a}, \frac{c-\sqrt{c^2-4ab}}{2a}\right)$$



Bifurcation diagram



$$a = b = 0.1$$
$$c \in (0,45]$$

References

Rössler, Otto E. "An equation for continuous chaos." *Physics Letters A* 57.5 (1976): 397-398.

This is the original paper. The Rossler attractor can be seen as a Lorentz attractor with one lobe. Differently from Lorentz, in Rossler there is just one non-linearity.

Letellier, Christophe, and Valérie Messager. "Influences on Otto E. Rössler's earliest paper on chaos." *International Journal of Bifurcation and Chaos* 20.11 (2010): 3585-3616.

Historical overview on the Rossler system and its main influence in physics.

Delage, Olivier, and Alain Bourdier. "Selection of Optimal Embedding Parameters Applied to Short and Noisy Time Series from Rössler System." *Journal of Modern Physics* 8.09 (2017): 1607.

All you need to smartly succeed in this TP.

Observability

$\dot{W} = \mathbf{A}W$ $\mathbf{A} \in \mathbb{R}^{m \times m}$ Jacobian $\mathbf{C} \in \mathbb{R}^{r \times m}$ measure matrix

$X = \mathbf{C}W$ $W \in \mathbb{R}^{m}$ state $X \in \mathbb{R}^{r}$ measure

The system is observable in X
if rank$(\mathbf{Q}) = m$

$$\mathbf{Q} = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \mathbf{CA}^2 \\ \vdots \\ \mathbf{CA}^{m-r} \end{bmatrix}$$

for Rossler $m = 3$ and observing
one coordinate $r = 1$

Observability in Rossler

$$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + ay \\ \dot{z} = b + z(x-c) \end{cases} \qquad \mathbf{A} = \begin{bmatrix} 0 & -1 & -1 \\ 1 & a & 0 \\ z & 0 & x-c \end{bmatrix} \qquad \mathbf{A}^2 = \begin{bmatrix} -1-z & -a & c-x \\ a & a^2-1 & -1 \\ z(x-c) & -z & -z+(x-c)^2 \end{bmatrix}$$

$$\mathbf{C}_y = [0 \quad 1 \quad 0] \qquad \mathbf{Q}_y = \begin{bmatrix} 0 & 1 & 0 \\ 1 & a & 0 \\ a & a^2-1 & -1 \end{bmatrix} \qquad \det(\mathbf{Q}_y) = 1$$

$$\mathbf{C}_x = [1 \quad 0 \quad 0] \qquad \mathbf{Q}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & -1 \\ -1-z & -a & c-x \end{bmatrix} \qquad \det(\mathbf{Q}_x) = 0 \textbf{ if } x = a + c$$

$$\mathbf{C}_z = [0 \quad 0 \quad 1] \qquad \mathbf{Q}_z = \begin{bmatrix} 0 & 0 & 1 \\ z & 0 & x-c \\ z(x-c) & -z & -z+(x-c)^2 \end{bmatrix} \qquad \det(\mathbf{Q}_z) = 0 \textbf{ if } z = 0$$

Goals for this TP

- Generate the data (time series) with $a = b = 0.2, c = 5.7$
- Learn the discrete or continuous dynamical system from the time series

$$W_{n+1} = NN(W_n) \qquad \dot{W} = NN(W)$$

70% of the note: recover statistics

- PDF
- Time correlations
- Spectral density
- …

These analysis are enough to reach the 70% of the note but more analyses can make the difference!

30% of the note: recover dynamics

- Equilibrium point (at least one)
- Lyapunov exponents (the largest one) $\lambda \approx 7 \times 10^{-2}$

differences

Discrete		Continuous
$W_{t+\Delta t} = M(W_t)$	dynamicsl system	$\dot{W} = F(W)$
$w_{t+\Delta t} = \mathbf{J} w_t$	linearized system	$\dot{w} = \mathbf{A} w$
$\mathbf{J}$	Jacobian	$\mathbf{A}$
	Note that $\mathbf{J} \approx e^{\mathbf{A}\Delta t}$	
$W_0 - M(W_0) = 0$	Equilibrium state	$F(W_0) = 0$

Constraints

One approach is enough

- Temporal embedding or memory, discrete system: **just ONE coordinate** y of course!

$$y_1 = NN(y_0, y_{-1}, y_{-2,...})$$
$$y_1 = NN(y_0, \dot{y}_0, \ddot{y}_0, \ldots)$$
$$y_1 = NN(y_0, [h_0])$$

h_0 memory in RNN

RNN, temporal convolutions, MLP with multi inputs $(y_0, y_{-1}, y_{-2}, \ldots)$

- No Temporal embedding and no memory, discrete system: **the whole state**

$$W_1 = NN(W_0)$$

- Continuous system: **no constraints**

$$\dot{W} = NN(W)$$

$\dot{W}$ can be recovered from the data! use finite differentiation. Once you have W and $\dot{W}$ is just a supervised learning. Otherwise you can ODE Net to infer the governing ODE system. ODE Net is available online.

Be smart

To find the equilibrium point and to evaluate the Lyapunov exponent, the Jacobian has to be computed. Introduce a penalization onto the sensibility of your model wrt the inputs!

$$loss = \|W - \hat{W}\| + \lambda(?)$$

$$\|\dot{W} - \hat{\dot{W}}\|$$

$$\|A - \hat{A}\|$$

$$\|\hat{A}\|_F$$

There is not an unique way to proceed! If you use a purely data drive approach (without explicitly introduce the true Jacobian in the loss function) will be appreciated.

Attention: with a RNN the total Jacobian needs to be computed!

Appendix D in: “Backpropagation Algorithms and Reservoir Computing in Recurrent Neural Networks for the Forecasting of Complex Spatiotemporal Dynamics”

deliver

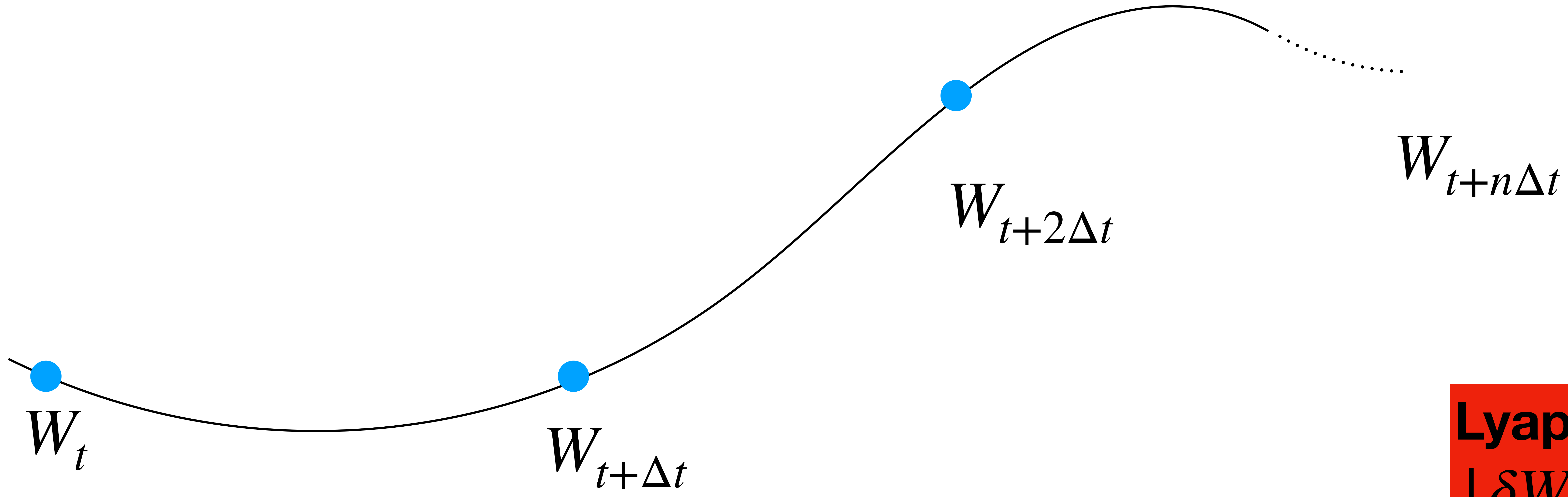
- small report ($\sim$4 pages)

  Justify the loss function (e.g. MSE, L_∞, penalizations, etc) and the performed analysis to validate your model

- codes to reproduce the pictures in the report (+ trained NN)
- code to generate a time series with your trained model (if discrete system let us know the marching Δt)

I need to run your code to generate e new time series knowing the Δt inbetween samples

Appendix



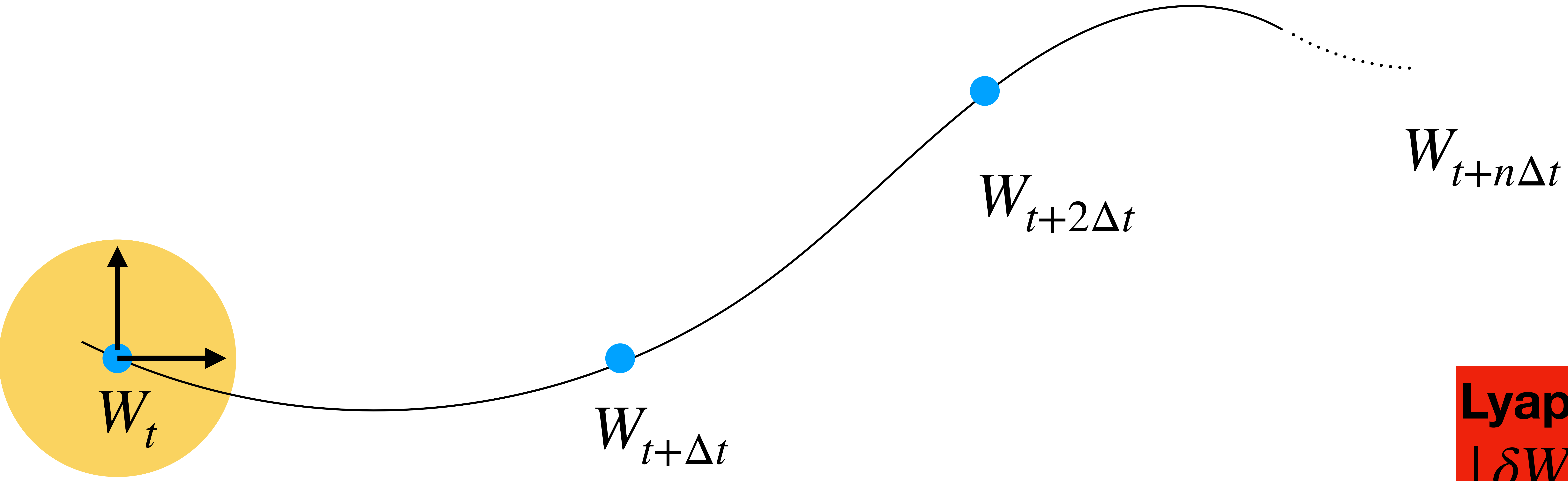
Lyapunov exponent:

$$|\delta W(t)| = e^{\lambda t}|\delta W_0|$$

where:

$$|\delta W_0| = |W_0 - W_0'|$$

Appendix



Lyapunov exponent:

$$|\delta W(t)| = e^{\lambda t}|\delta W_0|$$

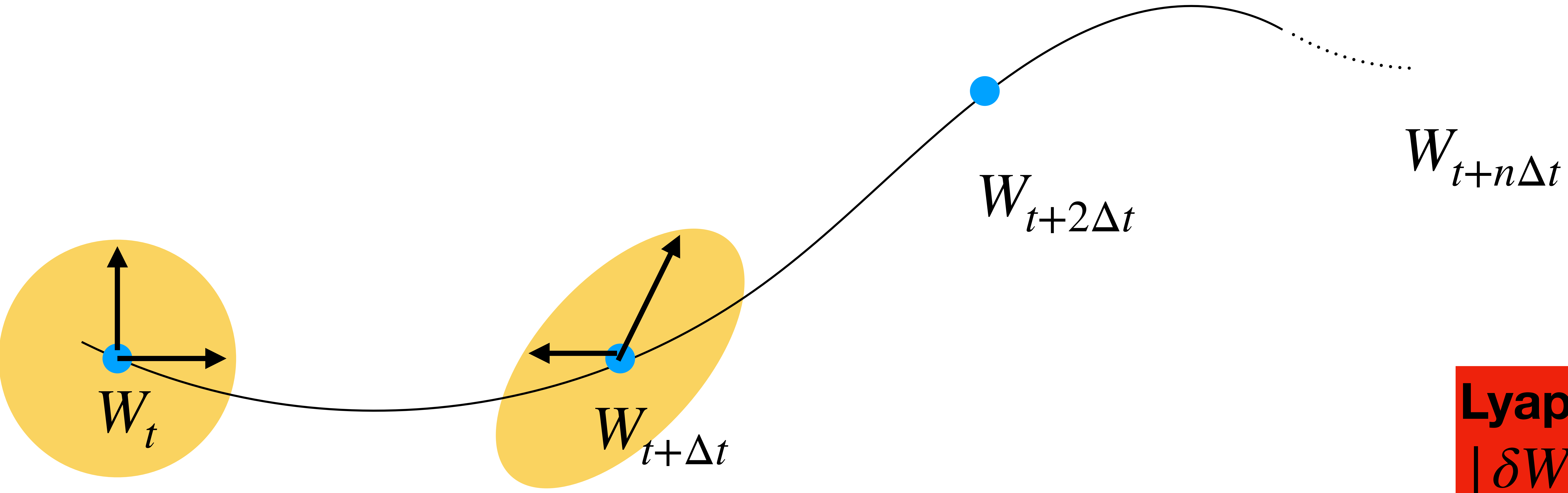
where:

$$|\delta W_0| = |W_0 - W_0'|$$

Initial guess with unitary energy

```
w = np.eye(n)
```

Appendix



Lyapunov exponent:

$$|\delta W(t)| = e^{\lambda t} |\delta W_0|$$

where:

$$|\delta W_0| = |W_0 - W_0'|$$

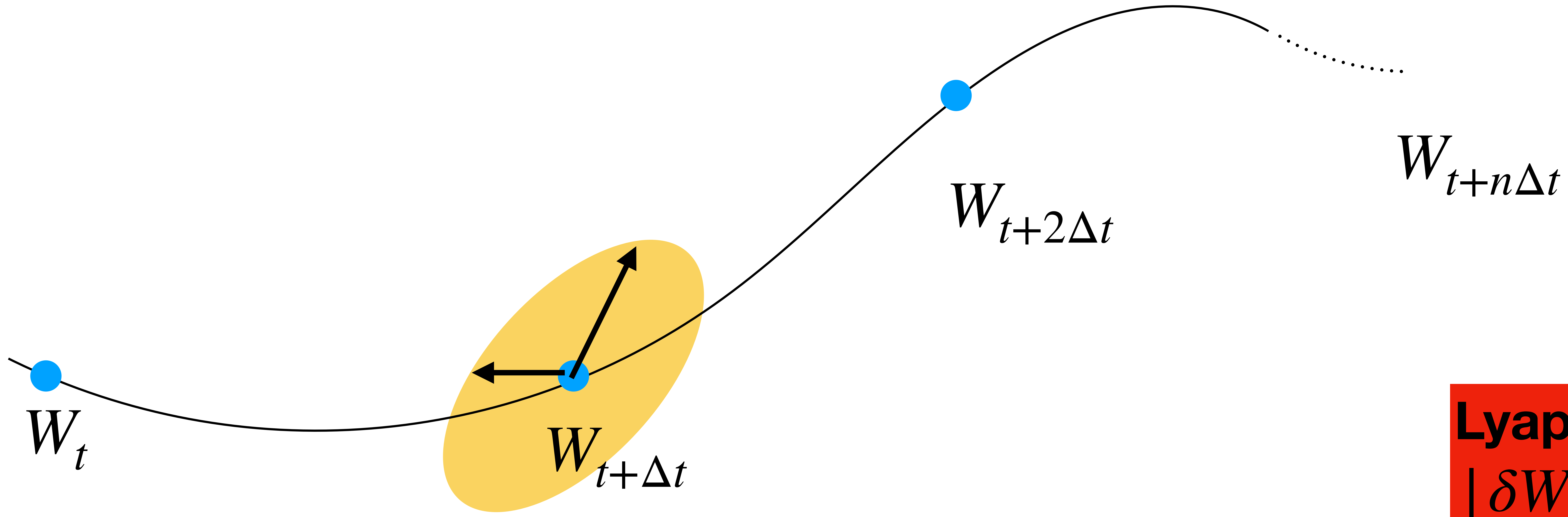
Initial guess with unitary energy

```
w = np.eye(n)
```

Evolution of the initial guess following the tangent trajectory

```
w_next = np.dot(expm(jacob * delta_t),w)
```

Appendix



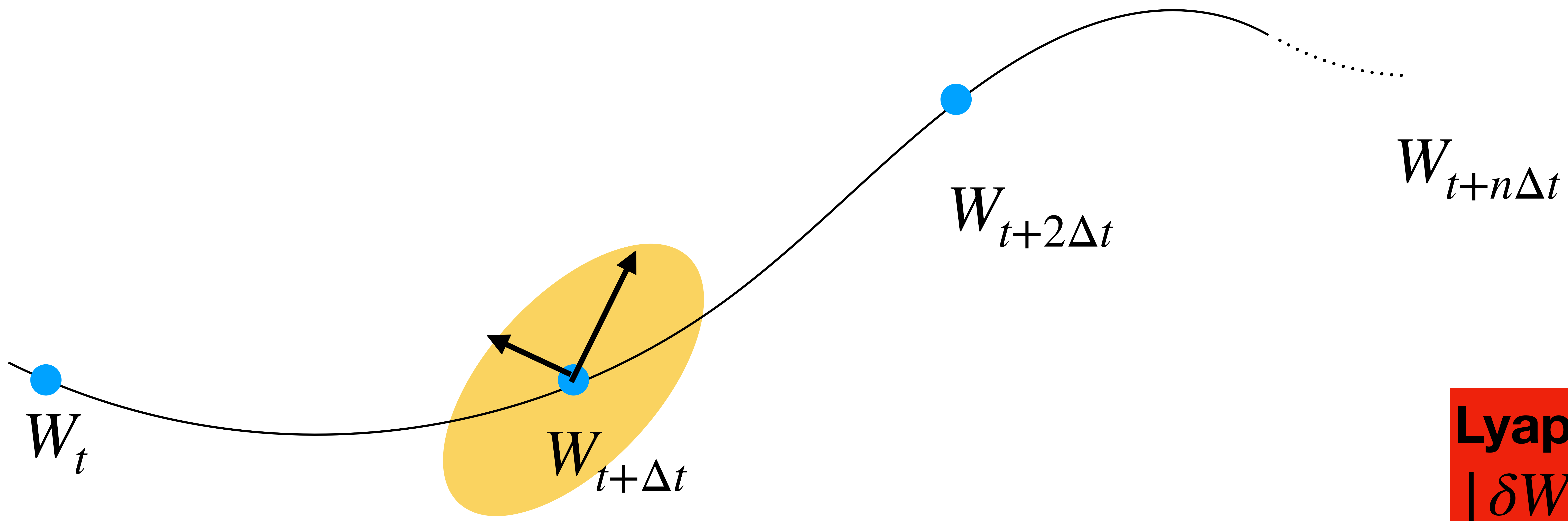
Lyapunov exponent:

$$|\delta W(t)| = e^{\lambda t} |\delta W_0|$$

where:

$$|\delta W_0| = |W_0 - W_0'|$$

Appendix



Lyapunov exponent:

$$|\delta W(t)| = e^{\lambda t}|\delta W_0|$$

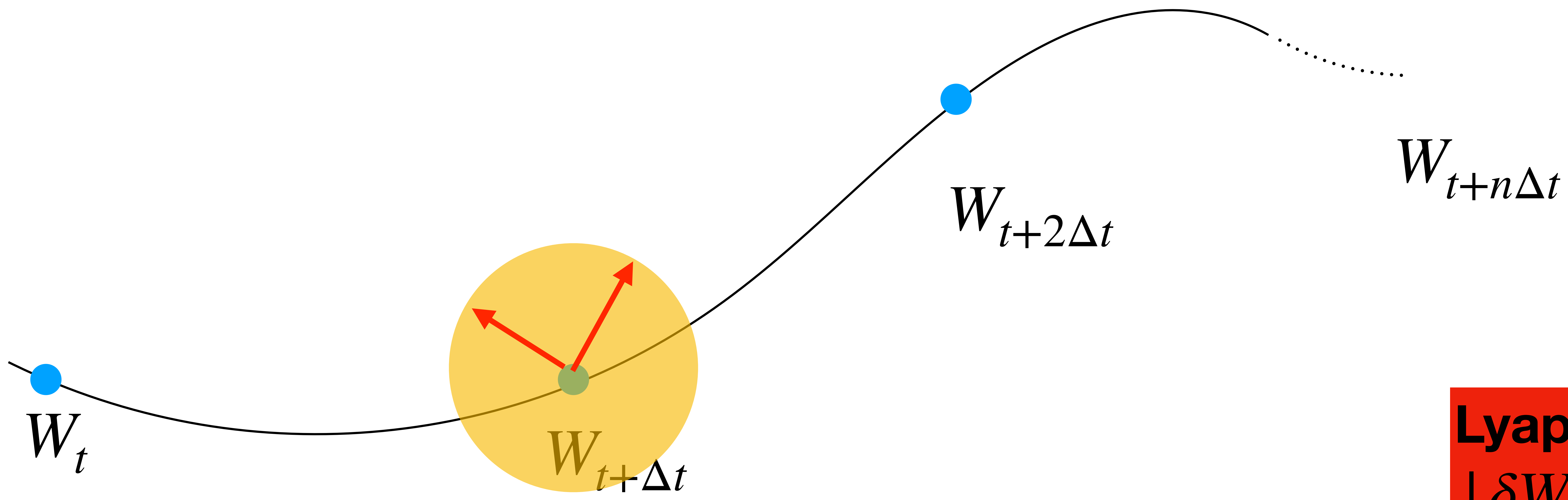
where:

$$|\delta W_0| = |W_0 - W_0'|$$

Evaluate the stretching and the new orthonormal base

```
w_next, r_next = qr(w_next)
```

Appendix



Lyapunov exponent:

$$|\delta W(t)| = e^{\lambda t} |\delta W_0|$$

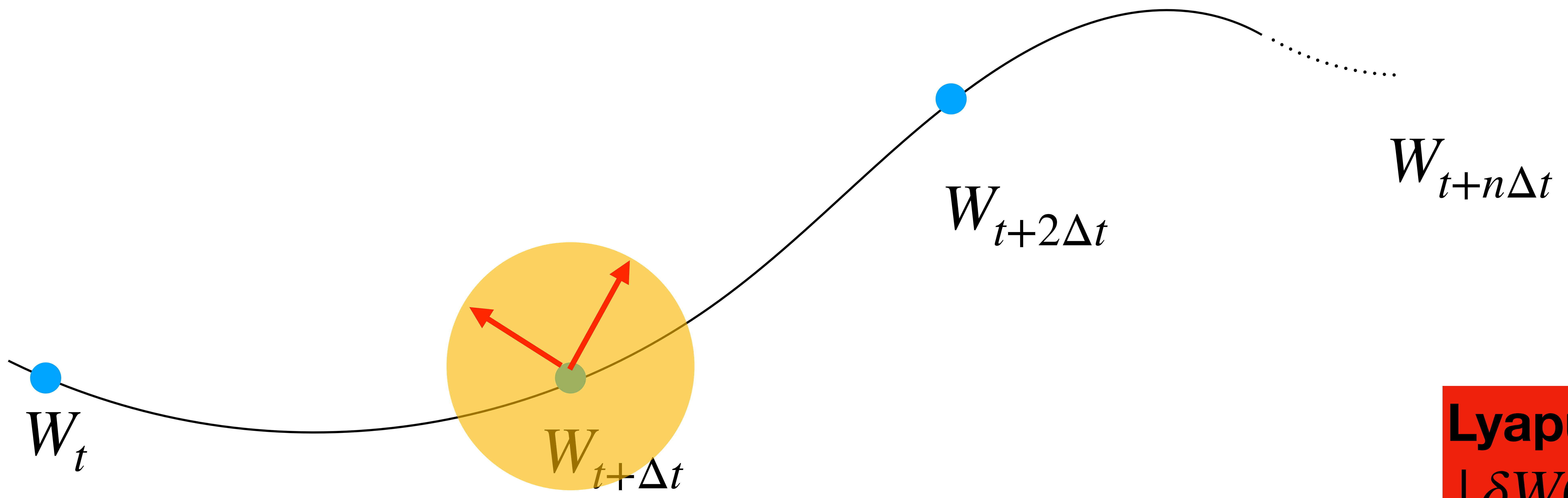
where:

$$|\delta W_0| = |W_0 - W_0'|$$

Evaluate the stretching and the new orthonormal base

```
w_next, r_next = qr(w_next)
```

Appendix



Lyapunov exponent:

$$|\delta W(t)| = e^{\lambda t} |\delta W_0|$$

where:

$$|\delta W_0| = |W_0 - W_0'|$$

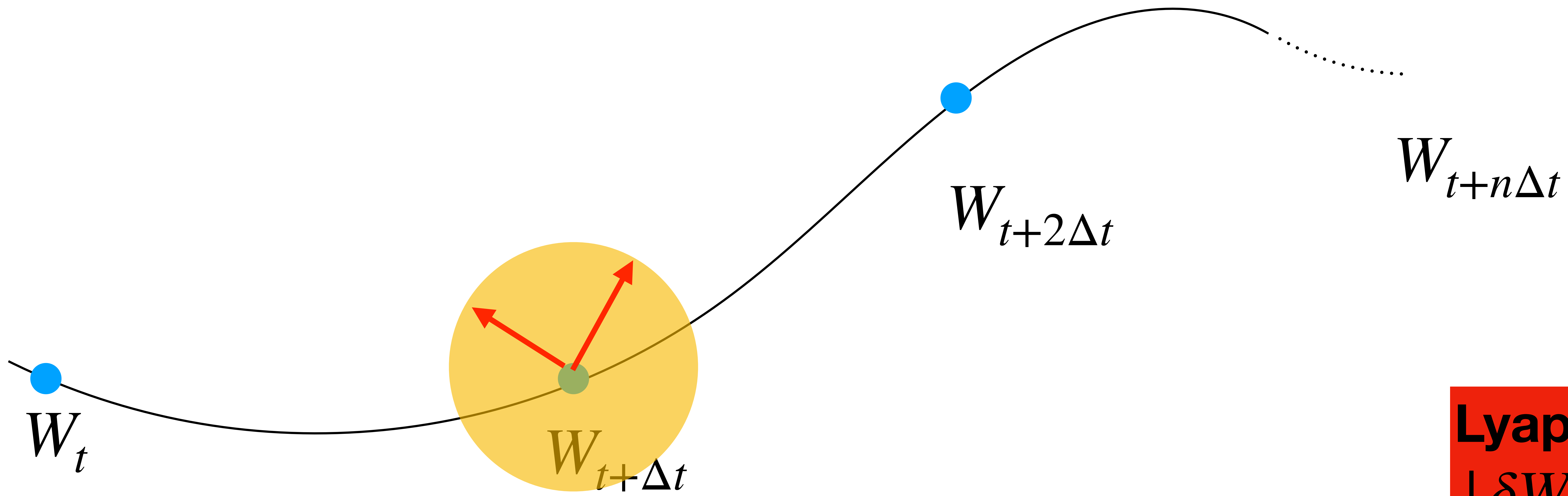
Evaluate the stretching and the new orthonormal base

```
w_next, r_next = qr(w_next)
```

Store the amplification factors

```
rs.append(r_next)
```

Appendix



Lyapunov exponent:

$$|\delta W(t)| = e^{\lambda t}|\delta W_0|$$

where:

$$|\delta W_0| = |W_0 - W_0'|$$

Evaluate the mean amplification

`mean(rs)` $\approx e^{\lambda_i t}$

`mean(log(rs))/t` $\approx \lambda_i$

PERSONAL CONSIDERATIONS

Training a RNN for a discrete system using just y coordinate is fast and safe. On the other hand recover the Jacobian is not straightforward.

Training a Neural Network to emulate the discrete Rossler system without temporal embedding is fast but you need to accurately design the loss function and the penalization. Recover the Jacobian is easy with automatic differeziation.

Training a Neural Network to recover the continuous Rossler system is hard. Exponentially more complicated ($\mathbf{J} \approx e^{\mathbf{A}\Delta t}$), an error on $\mathbf{A}$ is amplified in the solution propagation by $\mathbf{J}$. Moreover in the continuous space, cross trajectories are not allowed. BUT, if you are able to got it, the Jacobian is for free and almost exact.