Dynamical system formulation

The system is formulated in polar/spherical coordinates, thus cartesian data needs to be converted.
2D Dynamical System in polar coordinates $[\rho, \theta]$, where ρ is the polar radius and θ is the polar angle:

$$\mathbf{f}(\rho, \theta) = \begin{cases} \dot{\rho} = -\sqrt{2M}(\rho - \rho_0) \\ \dot{\theta} = R\mathrm{e}^{-4M^2(\rho-\rho_0)^2} \end{cases}$$

3D Dynamical System in spherical coordinates $[\rho, \phi, \theta]$, where ρ is the spherical radius, ϕ is the elevation angle (between x_3 and the x_1-x_2 plane) and θ is the azimuth angle (between x_1 and x_2):

$$\mathbf{f}(\rho, \phi, \theta) = \begin{cases} \dot{\rho} = -\sqrt{2M}(\rho - \rho_0) \\ \dot{\phi} = -\sqrt{2M}(\phi) \\ \dot{\theta} = R\mathrm{e}^{-4M^2(\rho-\rho_0)^2} \end{cases}$$

Parameters to be optimized (in red above):

- ρ_o : target radius; $\rho_o > 0$ for limit cycles, ρ_o = 0 for attractors. 1-dimensional.
- M : "mass"; $M > 0$; relates to speed to reach the target radius (higher speed for higher values) and to how fast the system "reacts" to the presence of the cycle/focus (rotation starts tight to target for higher values, while it starts rotating further away for lower values of M). 1-dimensional.
- R : rotation speed (only used for $\dot{\theta}$). 1-dimensional.

OPTIMIZATION

Least Squares minimizations:

$$\begin{aligned} 1. \quad & \min \| -\sqrt{2M}(\rho_{data} - \rho_0) - \dot{\rho}_{data} \|^2 \\ 2. \quad & \min \| R\, \mathrm{e}^{-4M^2(\rho_{data}-\rho_0)^2} - \dot{\theta}_{data} \|^2 \\ & u.c. \quad \rho_0 \geq 0, \quad M > 0 \end{aligned}$$

SCALING, SHIFTING AND ROTATION

For limit sets not centered to the origin, rotated on the plane/in space and/or with different scaling for various dimensions, transformations need to be applied **before** conversion to polar/spherical coordinates.
We aim to retrieve the *"original"* limit cycle, i.e. the circular, 0-centered cycle on the x_1-x_2 plane. We assume the *"original"* data was first rotated by a rotation matrix $Rrot$ (NxN orthogonal matrix, where N is the number of dimensions, either 2 or 3), then scaled by $\frac{1}{a}$ (a is a N-D vector) and finally shifted by x_0 (N-D vector). We thus plan to recover this *"original"* data $x_{original}$ by transforming the cartesian data x_{data}, as:

$$x_{original} = Rrot' * (a * (x_{data} + x_0))$$

(Note that the multiplication by the inverse of $Rrot$ is a left multiplication and x_{data} is expressed as a column vector. Since the matrix is orthogonal, $Rrot'$ is equal to the transpose.)
We then convert $x_{original}$ to polar/spherical coordinates, instead of x_{data}, and we will need to invert the equation above after calculating the DS to convert the final motion to the shifted, scaled & rotated space where x_{data} lives:

$$x_{original} = \frac{(Rrot * x_{data})}{a} - x_0$$

(Note that the multiplication by $Rrot$ is a left multiplication and x_{data} is expressed as a column vector.)
Added parameters to the optimization (included during conversion to polar/spherical coordinates):

- a : scaling along dimensions (for ellipses); $a \geq 1$. N-dimensional, with N being the number of dimensions of the data.
- x_0 : shift of the origin in space. N-dimensional.

Not included in the optimization above, but rather found in advance by singular value decomposition (for details please refer to the paper):

- $Rrot$: rotation matrix to bring the cycle on the x_1-x_2 plane. NxN-dimensional.