# paper

June 11, 2025

[1]:
```python
from pycandy import DiscreteDynamicalSystem as dds
from pycandy import PlotStyler
```

[2]:
```python
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
import matplotlib.gridspec as gridspec
import matplotlib as mpl
from matplotlib.colors import ListedColormap
import seaborn as sns
from string import ascii_lowercase
from joblib import Parallel, delayed
```

## 1 Basic system definition and simulation

[3]:
```python
dds.available_models()
```

[3]:
```
['standard map',
 'unbounded standard map',
 'henon map',
 'lozi map',
 'rulkov map',
 'logistic map',
 'standard nontwist map',
 'extended standard nontwist map',
 'leonel map',
 '4d symplectic map']
```

### 1.1 Standard map

[4]:
```python
ds = dds(model="standard map")
```

[5]:
```python
u = [0.05, 0.05] # initial condition
k = 1.5 # parameter for the standard map
total_time = 1000000 # total iteration time for each trajectory
```

[ ]:
```
%%time
trajectory = ds.trajectory(u, k, total_time)
```

```
35.8 ms ± 1.59 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

[18]:
```
trajectory.shape
```

[18]:
```
(1000000, 2)
```

[19]:
```
num_ic = 200 # number of initial conditions
np.random.seed(13) # for reproducibility
u = np.random.rand(num_ic, 2) # random initial conditions
k = 1.5 # parameter for the standard map
total_time = 100000 # total iteration time for each trajectory
```

[20]:
```
%%time
trajectories = ds.trajectory(u, k, total_time)
```

```
CPU times: user 1.35 s, sys: 58 ms, total: 1.4 s
Wall time: 356 ms
```

[21]:
```
trajectories.shape
```

[21]:
```
(20000000, 2)
```

[22]:
```
trajectories = trajectories.reshape(num_ic, total_time, 2)
```

[23]:
```
import seaborn as sns
colors = sns.color_palette("hls", num_ic)
np.random.seed(13) # for reproducibility
np.random.shuffle(colors) # shuffle the colors
```

[ ]:
```
fontsize = 15
plot_params(fontsize=fontsize, axes_linewidth=1.1)
fig, ax = plt.subplots(1, 2, sharex=True, sharey=True, figsize=(8, 3.5))
set_ticks_in(ax, pad_x=5)
ax[0].plot(trajectory[:, 0], trajectory[:, 1], 'ko', markersize=0.1,␣
 ↪markeredgewidth=0.0)

for i in range(num_ic):
    ax[1].plot(trajectories[i, :, 0], trajectories[i, :, 1], 'o',␣
 ↪color=colors[i], markersize=0.2, markeredgewidth=0.0)

xbox = 0.0067
ybox = 0.9457
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}

for i in range(2):
```

```
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
 ↪transform=ax[i].transAxes)

ax[0].set_xticks([0, 0.5, 1])
ax[0].set_yticks([0, 0.5, 1])
ax[0].set_xlim(0, 1)
ax[0].set_ylim(0, 1)
ax[0].set_xlabel(r"$x$")
ax[0].set_ylabel(r"$y$")
ax[1].set_xlabel(r"$x$")
plt.subplots_adjust(left=0.058, bottom=0.12, right=0.985, top=0.98, wspace=0.
 ↪07, hspace=0.2)
plt.savefig("fig1.png", dpi=400)
```

### 1.2 Hénon map

[3]:
```
ds = dds(model="henon map")
```

[4]:
```
info = ds.info
info["parameters"]
```

[4]:
```
['a', 'b']
```

[ ]:
```
u = [0.1, 0.1] # initial condition
a, b = 1.4, 0.3 # parameters for the Henon map
parameters = [a, b]
total_time = 1000000 # total iteration time for each trajectory
transient_time = 500000 # transient time for the Henon map
```

[ ]:
```
%%time
trajectory = ds.trajectory(u, parameters, total_time,␣
 ↪transient_time=transient_time)
```

```
CPU times: user 62.7 ms, sys: 1.47 ms, total: 64.2 ms
Wall time: 64.8 ms
```

[ ]:
```
trajectory.shape
```

[ ]:
```
(500000, 2)
```

[ ]:
```
u = [0.1, 0.1] # initial condition
# We are going to change the parameter a and keep b fixed at
b = 0.3
# We define the parameter array with only the b value because a is going to be␣
 ↪changed
parameters = b
# Define the parameter range
param_range = np.linspace(1, 1.4, 2500)
```

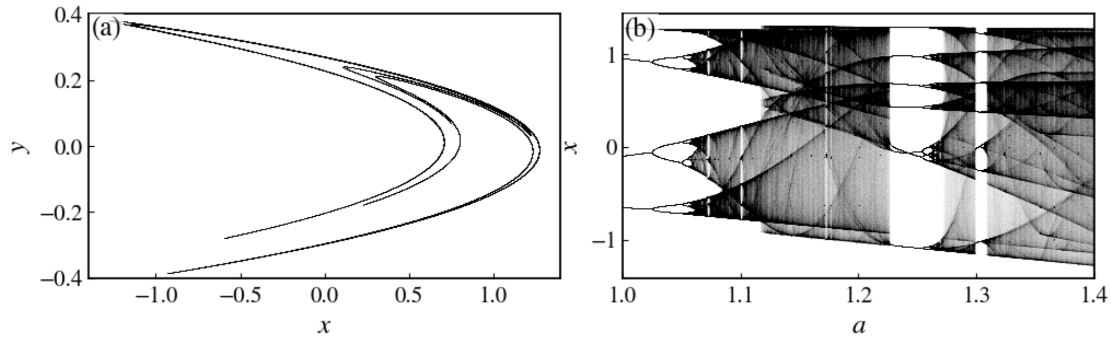
```
# Define which parameter will be changed
param_index = 0 # The parameter is the first one (parameters = [a, b])
# Define the total number of iterations (including the transient)
total_time = 8000
# Define the transient iterations
transient_time = 2000
```

[ ]:
```
%%time
param_values, bifurcation_diagram = ds.bifurcation_diagram(u, param_index,␣
↪param_range, total_time, parameters=parameters,␣
↪transient_time=transient_time)
```

```
CPU times: user 3.04 s, sys: 34.4 ms, total: 3.07 s
Wall time: 3.09 s
```

[ ]:
```
fontsize = 18
plot_params(fontsize=fontsize)
fig, ax = plt.subplots(1, 2, figsize=(10, 3))
set_ticks_in(ax, pad_x=5)
ax[0].plot(trajectory[:, 0], trajectory[:, 1], 'ko', markersize=0.2,␣
↪markeredgewidth=0.0)
ax[0].set_xlim(-1.4, 1.4)
ax[0].set_ylim(-.4, .4)
ax[0].set_xlabel(r"$x$")
ax[0].set_ylabel(r"$y$")

for i in range(bifurcation_diagram.shape[0]):
    ax[1].scatter(param_values[i] * np.ones_like(bifurcation_diagram[i, :]),␣
↪bifurcation_diagram[i, :], c="k", s=0.005, edgecolors="none")

ax[1].set_xlim(param_values.min(), param_values.max())
ax[1].set_xlabel(r"$a$")
ax[1].set_ylabel(r"$x$")

xbox = 0.006
ybox = 0.919
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}

for i in range(2):
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
↪transform=ax[i].transAxes)
plt.subplots_adjust(left=0.068, bottom=0.17, right=0.9875, top=0.975, wspace=0.
↪13)
plt.savefig("fig2.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```




# 2 Chaotic indicators

## 2.1 Lyapunov exponents

### 2.1.1 Final value

[5]:
```
from numba import njit
```

[6]:
```
@njit
def dakrm(u, parameters):
    k, a, gamma = parameters
    x, y = u

    y_new = (1 - gamma) * y + k * (np.sin(x) + a * np.sin(2 * x + np.pi / 2))
    x_new = (x + y_new) % (2 * np.pi)
    return np.array([x_new, y_new])

@njit
def dakrm_jacobian(u, parameters, *args):
    k, a, gamma = parameters
    x, y = u

    dFdx = k * (np.cos(x) + 2 * a * np.cos(2 * x + np.pi / 2))
    dFdy = 1 - gamma

    return np.array([
        [1 + dFdx, dFdy],
        [dFdx,     dFdy]
    ])
```

[7]:
```
ds = dds(mapping=dakrm, jacobian=dakrm_jacobian, system_dimension=2)
```

[8]:
```
from joblib import Parallel, delayed
```

[11]:
```python
# Initial condition
u = np.array([1.78, 0.0])
# Parameters
gamma = 0.8
grid_size = 1000
k = np.linspace(0, 30, grid_size)
a = np.linspace(0, 1, grid_size)
K, A = np.meshgrid(k, a)
# Total number of iterations (including the transient)
total_time = 10000
# Transient iterations
transient_time = 5000
```

[12]:
```python
k = 8
a = 0.47
gamma = 0.8
parameters = [k, a, gamma]
```

[16]:
```python
%%time
ds.lyapunov(u, parameters, total_time, transient_time=transient_time)
```

```
CPU times: user 5.92 ms, sys: 240  s, total: 6.16 ms
Wall time: 6.31 ms
```

[16]:
```
array([ 1.57224186, -3.18167977])
```

[14]:
```python
ds.period(u, parameters, total_time, transient_time=transient_time)
```

[14]:
```
2
```

[15]:
```python
k = 8
a = 0.6
gamma = 0.8
parameters = np.array([k, a, gamma])
```

[17]:
```python
%%time
ds.lyapunov(u, parameters, total_time, transient_time=transient_time)
```

```
CPU times: user 6.04 ms, sys: 301  s, total: 6.35 ms
Wall time: 6.44 ms
```

[17]:
```
array([ 1.57224186, -3.18167977])
```

[18]:
```python
ds.period(u, parameters, total_time, transient_time=transient_time)
```

[18]:
```
-1
```

[19]:
```python
ds = dds(mapping=dakrm, system_dimension=2)
```

[20]:
```python
# Initial condition
u = np.array([1.78, 0.0])
# Parameters
k = 8
a = 0.6
gamma = 0.8
parameters = np.array([k, a, gamma])
# Total number of iterations (including the transient)
total_time = 10000
# Transient iterations
transient_time = 5000
```

[21]:
```python
%%time
ds.lyapunov(u, parameters, total_time, transient_time=transient_time)
```

```
CPU times: user 433 ms, sys: 11.9 ms, total: 445 ms
Wall time: 447 ms
```

[21]:
```
array([ 1.5740678 , -3.18114158])
```

[ ]:
```python
%%time
lyapunov = np.array(Parallel(n_jobs=-1)(delayed(ds.lyapunov)(u, np.array([K[i,␣
↪j], A[i, j], gamma]), total_time, transient_time=1000) for i in␣
↪range(grid_size) for j in range(grid_size)))
```

[ ]:
```python
lyapunov.shape
```

[ ]:
```
(1000000, 2)
```

[ ]:
```python
%%time
period = np.array(Parallel(n_jobs=-1)(delayed(ds.period)(u, np.array([k_val,␣
↪a_val, gamma]), total_time=total_time, transient_time=transient_time) for␣
↪k_val, a_val in zip(K.flatten(), A.flatten())))
```

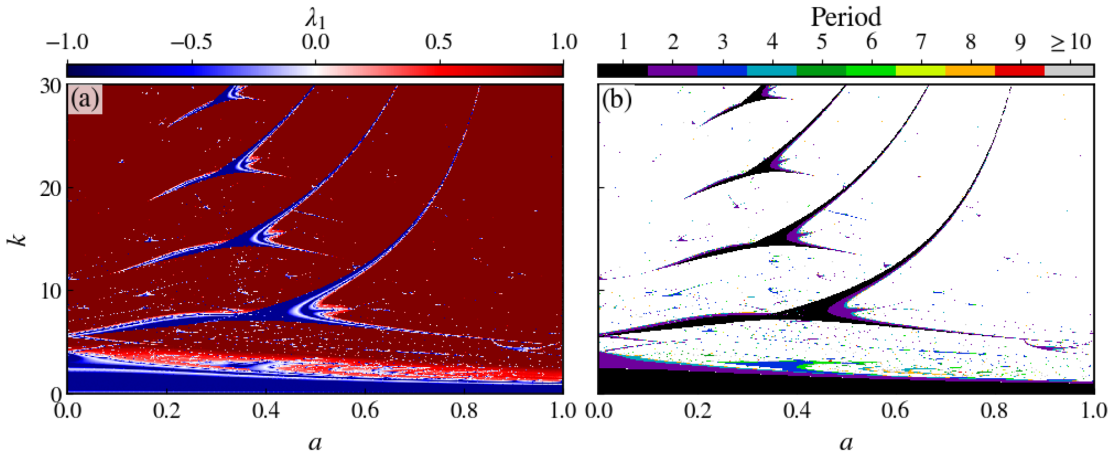
```
CPU times: user 55.3 s, sys: 1.97 s, total: 57.2 s
Wall time: 7min 2s
```

[ ]:
```python
import seaborn as sns
cmap = sns.color_palette("icefire", as_cmap=True)
```

[ ]:
```python
plot_params(fontsize=18)
fig, ax = plt.subplots(1, 2, figsize=(10, 4), sharex=True, sharey=True)
set_ticks_in(ax, pad_x=5)

hm = ax[0].pcolor(A, K, lyapunov[:, 0].reshape((grid_size, grid_size)),␣
↪cmap="seismic", vmin=-1, vmax=1)
plt.colorbar(hm, aspect=40, pad=0.02, label=r'$\lambda_1$', location="top")
ax[0].set_xlabel(r'$a$')
```

```python
ax[0].set_ylabel(r'$k$')


aux_period = np.asarray(period, dtype=np.float64).reshape(grid_size, grid_size)
aux_period[np.where(aux_period == -1)] = np.nan
cmap = plt.cm.nipy_spectral  # define the colormap
# create a list of colors
cmaplist = [cmap(i) for i in range(cmap.N)]
# create the new map
cmap = mpl.colors.LinearSegmentedColormap.from_list(
    'Custom cmap', cmaplist, cmap.N)
# define the bins and normalize
bounds = np.linspace(0.5, 10.5, 11)
norm = mpl.colors.BoundaryNorm(bounds, cmap.N)
hm = ax[1].pcolormesh(A, K, aux_period, cmap=cmap, norm=norm)
ticks = np.arange(1, 11, 1)
cbar = plt.colorbar(hm, ticks=ticks, label="Period", aspect=40, pad=0.02,␣
 ↪location="top")
ticks = list(ticks)
ticks[-1] = "$\\geq 10$"
cbar.ax.set_xticklabels(ticks)
cbar.ax.minorticks_off()
ax[1].set_xlabel(r'$a$')

xbox = 0.0058
ybox = 0.93
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}

for i in range(2):
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
 ↪transform=ax[i].transAxes)

plt.subplots_adjust(left=0.055, bottom=0.125, right=0.987, top=0.9675, wspace=0.
 ↪07)
plt.savefig("fig3.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```




### 2.1.2 Whole history

[3]:
```
ds = dds(model="standard map")
```

[4]:
```
k = 0.9
total_time = 1000000
u = np.array([[0.26, 0.0],
              [0.4, 0.],
              [0, 0.45],
              [0.1, 0.25],
              [0.1, 0.68],
              [0.06, 0.05],
              [0, 0.3],
              [0, 0.6],
              [0, 0.7]])
num_ic = u.shape[0] # number of initial conditions
```

[7]:
```
%%time
ts = ds.trajectory(u, total_time, parameters=k)
```

```
CPU times: user 932 ms, sys: 81.2 ms, total: 1.01 s
Wall time: 682 ms
```

[8]:
```
ts = ts.reshape(num_ic, total_time, 2)
```

[20]:
```
total_time = 100000000
sample_times = np.unique(np.logspace(np.log10(1), np.log10(total_time), 1000).
 ↪astype(int))
```

[21]:
```
%%time
```

```
lyapunovs = np.array(Parallel(n_jobs=-1)(delayed(ds.lyapunov)(u[i], total_time,␣
↪parameters=k, return_history=True, sample_times=sample_times) for i in␣
↪range(num_ic)))
```

```
CPU times: user 147 ms, sys: 50.4 ms, total: 197 ms
Wall time: 2min 29s
```

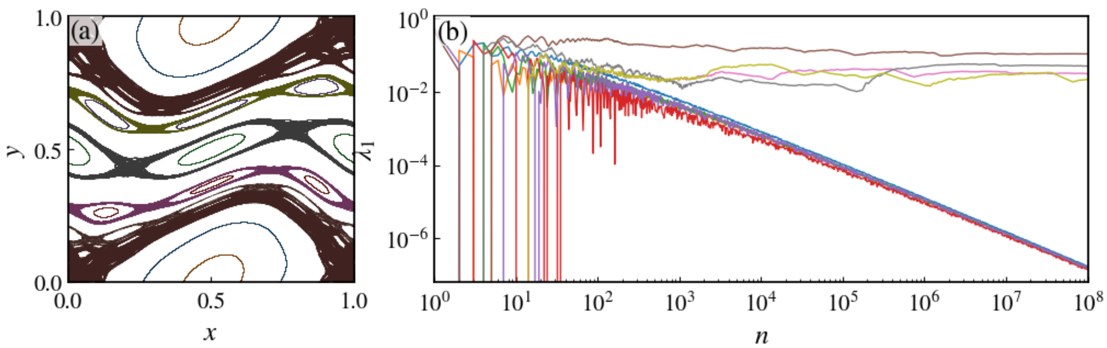
[22]:
```
lyapunovs.shape
```

[22]:
```
(9, 836, 2)
```

[23]:
```
ps = PlotStyler(fontsize=18, ticks_on_all_sides=False, markersize=0.2,␣
↪markeredgewidth=0)
ps.apply_style()
# Create figure
fig = plt.figure(figsize=(10, 3))
colors = sns.color_palette("tab10", num_ic)
# Create GridSpec with 1 row and 3 columns
gs = gridspec.GridSpec(1, 3)
ax1 = fig.add_subplot(gs[0, 0])
ax2 = fig.add_subplot(gs[0, 1:])
ax = np.array([ax1, ax2], dtype=object)
[ps.set_tick_padding(ax[i], pad_x=6) for i in range(ax.shape[0])]


for i in range(num_ic):
    ax[0].plot(ts[i, :, 0], ts[i, :, 1], 'o', color=colors[i])
    ax[1].plot(sample_times, lyapunovs[i, :, 0], '-', color=colors[i])

ax[0].set_xlim(0, 1)
ax[0].set_ylim(0, 1)
ax[0].set_xlabel(r"$x$")
ax[0].set_ylabel(r"$y$")
ax[0].set_xticks([0, 0.5, 1])
ax[0].set_yticks([0, 0.5, 1])

ax[1].set_xlabel(r"$n$")
ax[1].set_ylabel(r"$\lambda_1$")
ax[1].set_xlim(sample_times[0], sample_times[-1])
ax[1].set_yscale("log")
ax[1].set_xscale("log")

xbox = [0.01003, 0.004]
ybox = [0.919, 0.919]
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}

for i in range(2):
```

```
    ax[i].text(xbox[i], ybox[i], f"({ascii_lowercase[i]})", bbox=bbox,␣
 ↪transform=ax[i].transAxes)

plt.subplots_adjust(left=0.054, bottom=0.165, right=0.985, top=0.97, wspace=0.
 ↪28, hspace=0.2)

plt.savefig("fig4.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```




### 2.1.3 Finite-time Lyapunov exponent

[45]:
```
ds = dds(model="standard map")
```

[46]:
```
u = [0.5, 0.25]
nk = 5000
k = np.linspace(0, 5, nk)
total_time = 5000
```

[47]:
```
%%time
lypnvs_vs_k = np.array([ds.lyapunov(u, total_time, k[i]) for i in range(nk)])
```

```
CPU times: user 9.49 s, sys: 56 ms, total: 9.55 s
Wall time: 9.56 s
```

[48]:
```
u = [0.05, 0.05]
parameter = 1.5
# The total number of iterations for the FTRTE computation is
total_time = 100000000
# and the size of the windows is
finite_time = 200
```

[49]:
```
%%time
ftle = ds.finite_time_lyapunov(u, total_time, finite_time, parameters=parameter)
```

```
CPU times: user 38.8 s, sys: 204 ms, total: 39 s
Wall time: 39 s
```

[50]:
```python
total_time = 2000000
# and the size of the windows is
finite_time = 200
```

[51]:
```python
%%time
_, points = ds.finite_time_lyapunov(u, total_time, finite_time,␣
↪parameters=parameter, return_points=True)
```

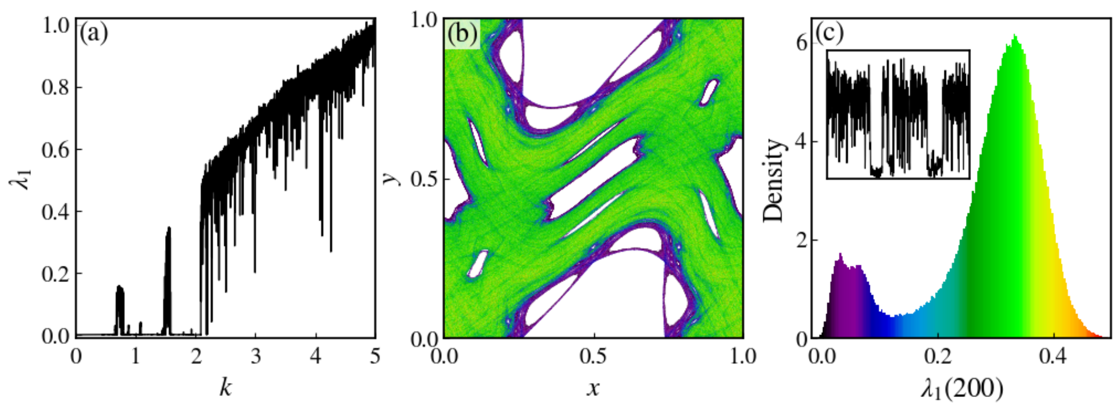
```
CPU times: user 839 ms, sys: 36.3 ms, total: 875 ms
Wall time: 846 ms
```

[52]:
```python
ts = ds.trajectory(points, finite_time, parameters=parameter)
ts = ts.reshape(points.shape[0], finite_time, 2)
```

[55]:
```python
fontsize = 18
ps = PlotStyler(fontsize=18, ticks_on_all_sides=False)
ps.apply_style()
fig, ax = plt.subplots(1, 3, figsize=(10, 3.5))
[ps.set_tick_padding(ax[i], pad_x=6) for i in range(ax.shape[0])]


ax[0].plot(k, lypnvs_vs_k[:, 0], "k")
ax[0].set_xlim(k.min(), k.max())
ax[0].set_xticks([0, 1, 2, 3, 4, 5])
ax[0].set_xlabel("$k$")
ax[0].set_ylabel(r"$\lambda_1$")
ax[0].set_ylim(-0.01, 1.02)

for i in range(points.shape[0]):
    ax[1].scatter(ts[i, :, 0], ts[i, :, 1], c=ftle[i, 0] * np.
↪ones(finite_time), cmap="nipy_spectral", s=0.05, edgecolors="none", vmin=0,␣
↪vmax=ftle[:, 0].max())

ax[1].set_xlim(0, 1)
ax[1].set_ylim(0, 1)
ax[1].set_xlabel(r"$x$")
ax[1].set_ylabel(r"$y$")
ax[1].set_xticks([0, 0.5, 1])
ax[1].set_yticks([0, 0.5, 1])

counts, bins, patches = ax[2].hist(ftle[:, 0], bins="auto", edgecolor='none',␣
↪density=True)
# Compute bin centers
bin_centers = 0.5 * (bins[:-1] + bins[1:])
```

```python
# Normalize bin centers for colormap
norm = plt.Normalize(0, bin_centers.max())
colormap = cm.nipy_spectral  # You can choose any colormap you like

# Apply color based on bin center (x position)
for center, patch in zip(bin_centers, patches):
    color = colormap(norm(center))
    patch.set_facecolor(color)

ax[2].set_xlim(bins[0], 0.5)
ax[2].set_xlabel(r"$\lambda_1(200)$")
ax[2].set_ylabel("Density")

ax_ins = ax[2].inset_axes([0.05, 0.5, 0.475, 0.4])  # [left, bottom, width,␣
↪height]

ii = np.arange(ftle.shape[0]) / 1e3
ax_ins.plot(ii, ftle[:, 0], "k", lw=0.75)
ax_ins.set_xlim(1, 2)
ax_ins.set_ylim(0, 0.5)
ax_ins.set_xticks([])
ax_ins.set_yticks([])

xbox = 0.0095
ybox = 0.9318
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}
for i in range(3):
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
↪transform=ax[i].transAxes)
plt.subplots_adjust(left=0.06, bottom=0.15, right=0.9975, top=0.975, wspace=0.
↪23, hspace=0.2)

plt.savefig("fig5.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```



### 2.2 Linear dependence index

[7]:
```
ds = dds(model="4D symplectic map")
```

[8]:
```
info = ds.info()
info["parameters"]
```

[8]: ['eps1', 'eps2', 'xi']

[9]:
```
u = np.array([[0.5, 0.0, 0.5, 0.0],
              [3.0, 0.0, 0.5, 0.0]])
parameters = np.array([0.5, 0.1, 0.001], dtype=np.float64)
total_time = int(1e6)
sample_times = np.unique(np.logspace(np.log10(1), np.log10(total_time), 5000).
 ↪astype(int))
```

[ ]:
```
%%time
lyapunovs = np.array(Parallel(n_jobs=-1)(delayed(ds.lyapunov)(u[i], total_time,␣
 ↪parameters=parameters, return_history=True, sample_times=sample_times) for i␣
 ↪in range(u.shape[0])))
```

```
CPU times: user 34.9 ms, sys: 109 ms, total: 144 ms
Wall time: 9.58 s
```

[11]:
```
lyapunovs.shape
```

[11]: (2, 3229, 4)

[12]:
```
k = [2, 3, 4]
total_time = int(1e5)
times = np.arange(1, total_time + 1)
ldi = np.zeros((u.shape[0], total_time, len(k)))
```

[ ]:
```
%%time
for i in range(len(k)):
    for j in range(u.shape[0]):
        ldi[j, :, i] = ds.LDI(u[j], total_time, parameters=parameters, k=k[i],␣
 ↪return_history=True)
```

```
/opt/anaconda3/lib/python3.12/site-packages/numba/core/utils.py:661:
NumbaExperimentalFeatureWarning: First-class function type feature is
experimental
  warnings.warn("First-class function type feature is experimental",
/opt/anaconda3/lib/python3.12/site-packages/numba/core/utils.py:661:
```

```
NumbaExperimentalFeatureWarning: First-class function type feature is
experimental
  warnings.warn("First-class function type feature is experimental",
```

```
CPU times: user 9.77 s, sys: 90.4 ms, total: 9.86 s
Wall time: 9.91 s
```

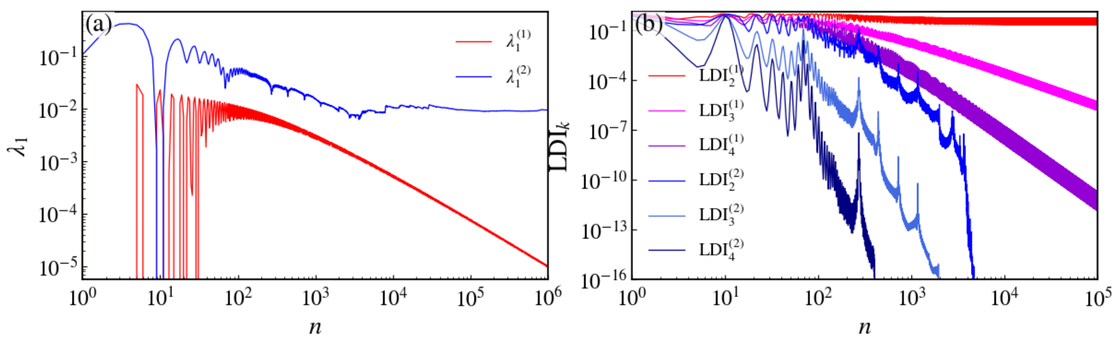
[14]:
```
ldi.shape
```

[14]:
```
(2, 100000, 3)
```

[15]:
```
plot_params(fontsize=20)
fig, ax = plt.subplots(1, 2, figsize=(12, 3.5))
set_ticks_in(ax, pad_x=6)


ax[0].plot(sample_times, lyapunovs[0, :, 0], '-', color="red", lw=0.9,␣
 ↪label=r"$\lambda_1^{(1)}$")
ax[0].plot(sample_times, lyapunovs[1, :, 0], '-', color="blue", lw=0.9,␣
 ↪label=r"$\lambda_1^{(2)}$")

colors = [['r', 'r', 'fuchsia', 'darkviolet'],
          ['b', 'b', 'royalblue', 'navy']]
for i in range(u.shape[0]):
    ax[1].plot(times, ldi[i, :, 0], "-", color=colors[i][1], lw=0.9,␣
 ↪label=f"LDI$_2^{{({i + 1})}}$")
    ax[1].plot(times, ldi[i, :, 1], "-", color=colors[i][2], lw=0.9,␣
 ↪label=f"LDI$_3^{{({i + 1})}}$")
    ax[1].plot(times, ldi[i, :, 2], "-", color=colors[i][3], lw=0.9,␣
 ↪label=f"LDI$_4^{{({i + 1})}}$")

ax[0].legend(loc="upper right", frameon=False)
ax[1].legend(loc="lower left", frameon=False)

ax[0].set_xlim(sample_times[0], sample_times[-1])
ax[0].set_xlabel(r"$n$")
ax[0].set_ylabel(r"$\lambda_1$")
ax[0].set_xscale("log")
ax[0].set_yscale("log")

ax[1].set_xlim(times[0], times[-1])
ax[1].set_ylim(1e-16, np.sqrt(2))
ax[1].set_xlabel(r"$n$")
ax[1].set_ylabel(r"LDI$_k$")
ax[1].set_xscale("log")
ax[1].set_yscale("log")
```

```
xbox = 0.0049
ybox = 0.9265
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}

for i in range(2):
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
 ↪transform=ax[i].transAxes)

plt.subplots_adjust(left=0.0675, bottom=0.16, right=0.985, top=0.99, wspace=0.
 ↪18, hspace=0.2)
plt.savefig("fig5.png", dpi=400)
```

<Figure size 640x480 with 0 Axes>




[16]:
```
total_time = 1000
nruns = 500
from time import time
```

[ ]:
```
exe_times = []
for _ in range(nruns):
    time_ini = time()
    ds.LDI(u[0], total_time, parameters=parameters, k=2)
    time_end = time()
    exe_times.append(time_end - time_ini)
LDI_time = np.mean(exe_times)
print(f"Execution time for LDI: {np.mean(exe_times):.5f} +- {np.std(exe_times):.
 ↪5f} seconds")
```

```
Execution time for LDI: 0.02749 +- 0.00064 seconds
```

[ ]:
```
exe_times = []
for _ in range(nruns):
    time_ini = time()
    ds.SALI(u[0], total_time, parameters=parameters)
    time_end = time()
```

```
    exe_times.append(time_end - time_ini)
SALI_time = np.mean(exe_times)
print(f"Execution time for SALI: {np.mean(exe_times):.6f} +- {np.std(exe_times):
 ↪.6f} seconds")
```

```
Execution time for SALI: 0.002777 +- 0.041096 seconds
```

[19]:
```
LDI_time, SALI_time, LDI_time / SALI_time
```

[19]:
```
(0.027492702960968018, 0.002777045249938965, 9.899983790891511)
```

### 2.3 Weighted Birkhoff averagas

[3]:
```
ds = dds(model="standard map")
```

[ ]:
```
k = 1.5
y = np.linspace(0, 1, 20001)
x = 0.5 * np.ones_like(y)
u = np.array([x, y]).T
total_time = 10000
```

[ ]:
```
%%time
dig = np.array([ds.dig(u[i], total_time, parameters=k) for i in range(u.
 ↪shape[0])])
```

```
/Users/mrolims/Library/CloudStorage/Dropbox/Física/Pesquisa/pycandy/src/pycandy
/dynamical_indicators.py:528: RuntimeWarning: divide by zero encountered in
log10
  return - np.log10(abs(WB0 - WB1))
```

```
CPU times: user 8.47 s, sys: 179 ms, total: 8.65 s
Wall time: 8.66 s
```

[ ]:
```
%%time
dig2 = np.array([ds.dig(u[i], total_time, parameters=k, func=lambda x: np.sin(2␣
 ↪* np.pi * x[:, 0])) for i in range(u.shape[0])])
```

```
CPU times: user 8.56 s, sys: 180 ms, total: 8.74 s
Wall time: 8.76 s
```

[ ]:
```
%%time
dig3 = np.array([ds.dig(u[i], total_time, parameters=k, func=lambda x: np.sin(2␣
 ↪* np.pi * (x[:, 0] + x[:, 1]))) for i in range(u.shape[0])])
```

```
CPU times: user 8.64 s, sys: 138 ms, total: 8.78 s
Wall time: 8.79 s
```

[231]:
```
grid_size = 1000
x = np.linspace(0, 1, grid_size)
y = np.linspace(0, 1, grid_size)
```

```
X, Y = np.meshgrid(x, y)
u = np.array([X.flatten(), Y.flatten()]).T
k = 1.5
N = 10000
```

[ ]:
```
%%time
dig = np.array(Parallel(n_jobs=-1)(delayed(ds.dig)(u[i], total_time,␣
↪parameters=k) for i in range(u.shape[0])))
```

```
/Users/mrolims/Library/CloudStorage/Dropbox/Física/Pesquisa/pycandy/src/pycandy
/dynamical_indicators.py:528: RuntimeWarning: divide by zero encountered in
log10
  return - np.log10(abs(WB0 - WB1))
/Users/mrolims/Library/CloudStorage/Dropbox/Física/Pesquisa/pycandy/src/pycandy
/dynamical_indicators.py:528: RuntimeWarning: divide by zero encountered in
log10
  return - np.log10(abs(WB0 - WB1))
/Users/mrolims/Library/CloudStorage/Dropbox/Física/Pesquisa/pycandy/src/pycandy
/dynamical_indicators.py:528: RuntimeWarning: divide by zero encountered in
log10
  return - np.log10(abs(WB0 - WB1))
/Users/mrolims/Library/CloudStorage/Dropbox/Física/Pesquisa/pycandy/src/pycandy
/dynamical_indicators.py:528: RuntimeWarning: divide by zero encountered in
log10
  return - np.log10(abs(WB0 - WB1))
```

```
CPU times: user 28.2 s, sys: 877 ms, total: 29 s
Wall time: 1min 36s
```

[233]:
```
dig = dig.reshape(grid_size, grid_size)
```

[ ]:
```
%%time
dig2 = np.array(Parallel(n_jobs=-1)(delayed(ds.dig)(u[i], total_time,␣
↪parameters=k, func=lambda x: np.sin(2 * np.pi * x[:, 0])) for i in range(u.
↪shape[0])))
```

```
CPU times: user 4min 37s, sys: 2.84 s, total: 4min 40s
Wall time: 4min 40s
```

[180]:
```
dig2 = dig2.reshape(grid_size, grid_size)
```

[181]:
```
dig2.shape
```

[181]:
```
(1000, 1000)
```

[ ]:
```
%%time
dig3 = np.array(Parallel(n_jobs=-1)(delayed(ds.dig)(u[i], total_time,␣
↪parameters=k, func=lambda x: np.sin(2 * np.pi * (x[:, 0] + x[:, 1]))) for i␣
↪in range(u.shape[0])))
```

```
CPU times: user 5min 13s, sys: 3.75 s, total: 5min 17s
Wall time: 5min 18s
```

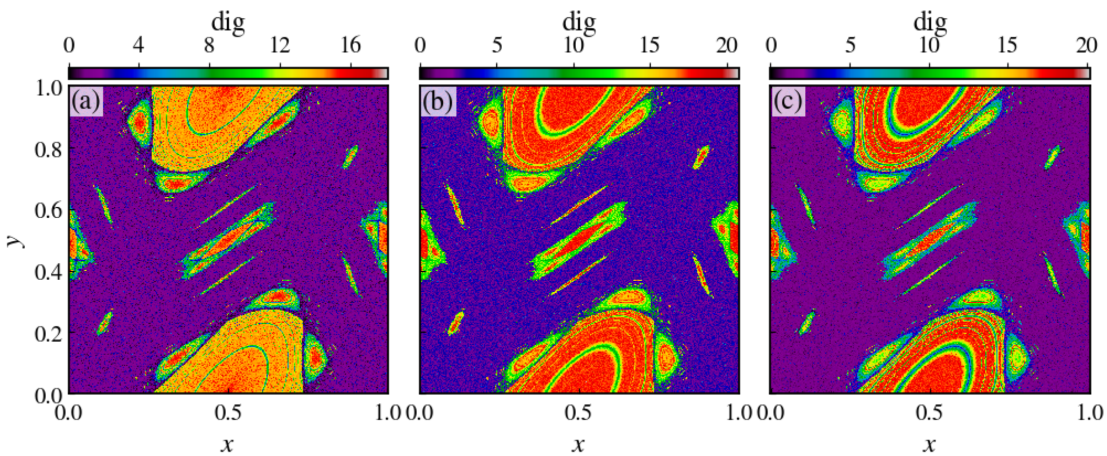
[183]:
```python
dig3 = dig3.reshape(grid_size, grid_size)
```

[234]:
```python
# Remove inf from dig: substitute with the mean of the neighbors
def remove_inf(dig):
    dig[np.isinf(dig)] = np.nan
    for i in range(dig.shape[0]):
        for j in range(dig.shape[1]):
            if np.isnan(dig[i, j]):
                neighbors = []
                if i > 0:
                    neighbors.append(dig[i - 1, j])
                if i < dig.shape[0] - 1:
                    neighbors.append(dig[i + 1, j])
                if j > 0:
                    neighbors.append(dig[i, j - 1])
                if j < dig.shape[1] - 1:
                    neighbors.append(dig[i, j + 1])
                dig[i, j] = np.nanmean(neighbors)
    return dig
dig_ri = remove_inf(dig)
dig2_ri = remove_inf(dig2)
dig3_ri = remove_inf(dig3)
```

[241]:
```python
plot_params(fontsize=18)
fig, ax = plt.subplots(1, 3, figsize=(10, 4), sharex=True, sharey=True)
set_ticks_in(ax, pad_x=6)
cmap = "nipy_spectral"
# cmap = sns.color_palette("magma", as_cmap=True)
hm = ax[0].pcolor(X, Y, dig_ri, cmap=cmap, vmin=0, vmax=dig_ri.max())
cbar = plt.colorbar(hm, aspect=30, pad=0.02, label="dig", location="top")
cbar.ax.set_xticks([0, 4, 8, 12, 16])
ax[0].set_xlabel(r"$x$")
ax[0].set_ylabel(r"$y$")

hm = ax[1].pcolor(X, Y, dig2_ri, cmap=cmap, vmin=0, vmax=dig2_ri.max())
cbar = plt.colorbar(hm, aspect=30, pad=0.02, label="dig", location="top")
cbar.ax.set_xticks([0, 5, 10, 15, 20])
ax[1].set_xlabel(r"$x$")
hm = ax[2].pcolor(X, Y, dig3_ri, cmap=cmap, vmin=0, vmax=dig3_ri.max())
cbar = plt.colorbar(hm, aspect=30, pad=0.02, label="dig", location="top")
cbar.ax.set_xticks([0, 5, 10, 15, 20])
ax[2].set_xlabel(r"$x$")

xbox = 0.009
```

```
ybox = 0.9298
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}

for i in range(3):
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
 ↪transform=ax[i].transAxes)

plt.subplots_adjust(left=0.06, bottom=0.13, right=0.985, top=0.97, wspace=0.1,␣
 ↪hspace=0.2)

plt.savefig("fig6.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```




```
[ ]:
```

## 2.4 Recurrence time entropy

```
[3]: ds = dds(model="standard map")
```

### 2.4.1 Recurrence matrix

```
[4]: u = [[0.05, 0.05],
          [0.35, 0.0],
          [0.42, 0.2]]
     k = 1.5
     total_time = 1000
```

```
[ ]: %%time
     recmats = [ds.recurrence_matrix(u[i], k, total_time) for i in range(len(u))]
```

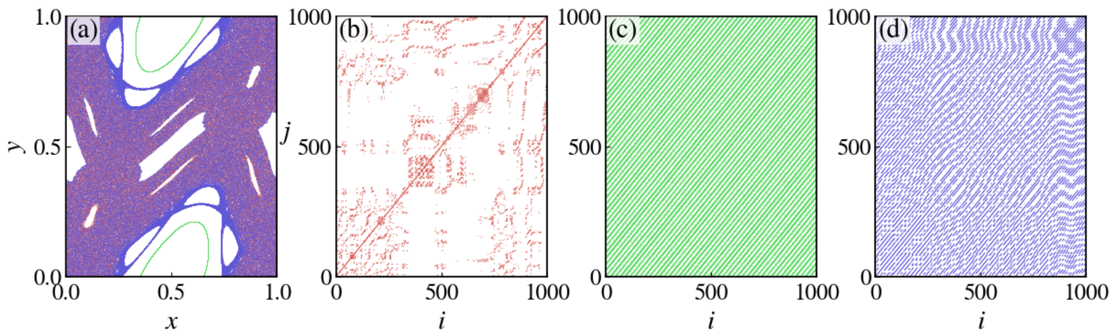
```
CPU times: user 3.33 ms, sys: 1.43 ms, total: 4.76 ms
Wall time: 4.04 ms
```

[ ]:
```
N = 250000
ts = ds.trajectory(u, total_time, parameters=k)
ts = ts.reshape(len(u), N, 2)
```

[14]:
```
plot_params(fontsize=21)
fig, ax = plt.subplots(1, 4, figsize=(12, 3.5))
set_ticks_in(ax, pad_x=5)


colors = sns.color_palette("hls", len(u))


for i in range(len(u)):
    # ax[0].scatter(u[i][0], u[i][1], c=colors[i], s=10, edgecolors="none")
    ax[0].plot(ts[i, :, 0], ts[i, :, 1], 'o', markersize=0.3, markeredgewidth=0.
 ↪0, color=colors[i])
    x = np.where(recmats[i] == 1)[0]
    y = np.where(recmats[i] == 1)[1]

    ax[1 + i].scatter(x, y, s=0.5, color=colors[i], edgecolors="none")
    ax[1 + i].set_xlim(0, total_time)
    ax[1 + i].set_ylim(0, total_time)
    ax[1 + i].set_xlabel(r"$i$")
    ax[i + 1].set_yticks([0, 500, 1000])
    ax[i + 1].set_xticks([0, 500, 1000])
ax[0].set_xlim(0, 1)
ax[0].set_ylim(0, 1)
ax[0].set_xlabel(r"$x$")
ax[0].set_ylabel(r"$y$")
ax[0].set_xticks([0, 0.5, 1])
ax[0].set_yticks([0, 0.5, 1])
label = ax[1].set_ylabel(r"$j$", rotation=0, labelpad=0.1)
# label.set_position((1, 0.6))  # (x, y) in axis coordinates
# ax[2].set_yticklabels([])
# ax[3].set_yticklabels([])

xbox = 0.012
ybox = 0.921
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}
for i in range(4):
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
 ↪transform=ax[i].transAxes)
```

```
plt.subplots_adjust(left=0.0525, bottom=0.161, right=0.98, top=0.972, wspace=0.
    ↪28, hspace=0.2)

plt.savefig("fig8.png", dpi=400)
```

<Figure size 640x480 with 0 Axes>




[6]:
```
u = [0.5, 0.25]
nk = 5000
k = np.linspace(0, 5, nk)
total_time = 5000
```

[7]:
```
%%time
rte = Parallel(n_jobs=-1)(delayed(ds.recurrence_time_entropy)(u, total_time,␣
    ↪parameters=k[i]) for i in range(k.shape[0]))
rte = np.array(rte)
```

CPU times: user 2.8 s, sys: 148 ms, total: 2.95 s
Wall time: 19.2 s

[9]:
```
u = [0.05, 0.05]
parameter = 1.5
# The total number of iterations for the FTRTE computation is
total_time = 100000000
# and the size of the windows is
finite_time = 200
```

[11]:
```
%%time
ftrte = ds.finite_time_recurrence_time_entropy(u, total_time, finite_time,␣
    ↪parameters=parameter)
```

CPU times: user 34.8 s, sys: 264 ms, total: 35.1 s
Wall time: 35.2 s

[39]:
```python
total_time = 2000000
# and the size of the windows is
finite_time = 200
```

[40]:
```python
%%time
_, points = ds.finite_time_recurrence_time_entropy(u, total_time, finite_time,␣
↪parameters=parameter, return_points=True)
```

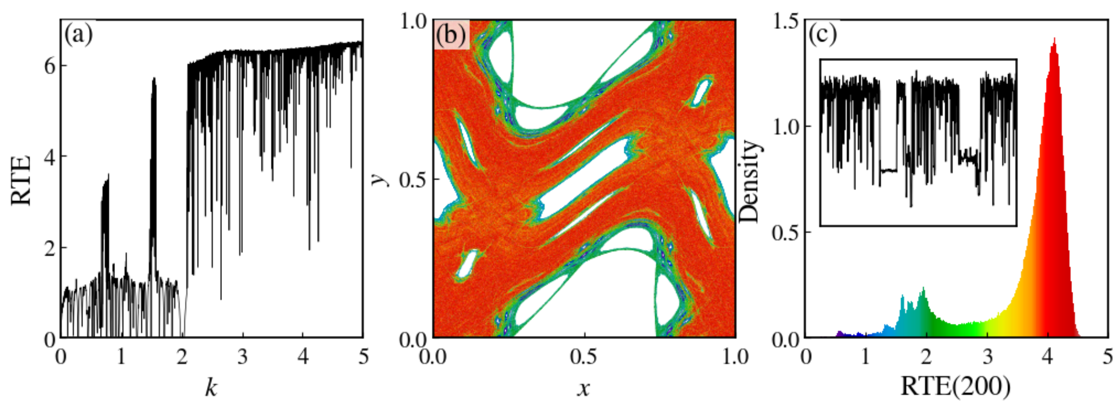
```
CPU times: user 796 ms, sys: 77.6 ms, total: 874 ms
Wall time: 804 ms
```

[41]:
```python
ts = ds.trajectory(points, finite_time, parameters=parameter)
ts = ts.reshape(len(points), finite_time, 2)
```

[44]:
```python
fontsize = 18
ps = PlotStyler(fontsize=fontsize, ticks_on_all_sides=False)
ps.apply_style()
fig, ax = plt.subplots(1, 3, figsize=(10, 3.5))
[ps.set_tick_padding(ax[i], pad_x=6) for i in range(ax.shape[0])]

ax[0].plot(k, rte, "k", lw=0.5)
ax[0].set_xlim(k.min(), k.max())
ax[0].set_xticks([0, 1, 2, 3, 4, 5])
ax[0].set_ylim(0, 7)
ax[0].set_xlabel(r"$k$")
ax[0].set_ylabel(r"RTE")

for i in range(points.shape[0]):
    ax[1].scatter(ts[i, :, 0], ts[i, :, 1], c=ftrte[i] * np.ones(finite_time),␣
↪cmap="nipy_spectral", s=0.05, edgecolors="none", vmin=0, vmax=ftrte.max())

ax[1].set_xlim(0, 1)
ax[1].set_ylim(0, 1)
ax[1].set_xlabel(r"$x$")
ax[1].set_ylabel(r"$y$")
ax[1].set_xticks([0, 0.5, 1])
ax[1].set_yticks([0, 0.5, 1])

counts, bins, patches = ax[2].hist(ftrte, bins="auto", edgecolor='none',␣
↪density=True)
# Compute bin centers
bin_centers = 0.5 * (bins[:-1] + bins[1:])

# Normalize bin centers for colormap
norm = plt.Normalize(0, bin_centers.max())
colormap = cm.nipy_spectral  # You can choose any colormap you like
```

```python
# Apply color based on bin center (x position)
for center, patch in zip(bin_centers, patches):
    color = colormap(norm(center))
    patch.set_facecolor(color)

ax[2].set_xlim(bins[0], bins[-1])
ax[2].set_xlabel(r"RTE(200)")
ax[2].set_ylabel("Density")
ax[2].set_xticks([0, 1, 2, 3, 4, 5])
ax[2].set_yticks([0, 0.5, 1, 1.5])

ax_ins = ax[2].inset_axes([0.05, 0.35, 0.65, 0.525])  # [left, bottom, width,␣
↪height]
ii = np.arange(ftrte.shape[0]) / 1e3
ax_ins.plot(ii, ftrte, "k", lw=0.75)
ax_ins.set_xlim(1, 2)
ax_ins.set_xticks([])
ax_ins.set_yticks([])

xbox = 0.0095
ybox = 0.9318
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}
for i in range(3):
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
↪transform=ax[i].transAxes)
plt.subplots_adjust(left=0.045, bottom=0.15, right=0.995, top=0.975, wspace=0.
↪23, hspace=0.2)
plt.savefig("fig9.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```



## 2.5 Hurst exponent

[56]:
```
ds = dds(model="standard map")
```

[60]:
```
u = [0.5, 0.25]
nk = 5000
k = np.linspace(0, 5, nk)
total_time = 5000
```

[61]:
```
%%time
HE = Parallel(n_jobs=-1)(delayed(ds.hurst_exponent)(u, total_time,␣
↪parameters=k[i]) for i in range(k.shape[0]))
HE = np.array(HE)
```

```
CPU times: user 5.19 s, sys: 227 ms, total: 5.42 s
Wall time: 1min 9s
```

[62]:
```
u = [0.05, 0.05]
parameter = 1.5
# The total number of iterations for the FTRTE computation is
total_time = 100000000
# and the size of the windows is
finite_time = 200
```

[64]:
```
%timeit
ftHE = ds.finite_time_hurst_exponent(u, total_time, finite_time,␣
↪parameters=parameter)
```

[75]:
```
ftHE_avg = (ftHE[:, 0] + ftHE[:, 1]) / 2
```

[91]:
```
total_time = 2000000
# and the size of the windows is
finite_time = 200
```

[92]:
```
%%time
_, points = ds.finite_time_hurst_exponent(u, total_time, finite_time,␣
↪parameters=parameter, return_points=True)
```

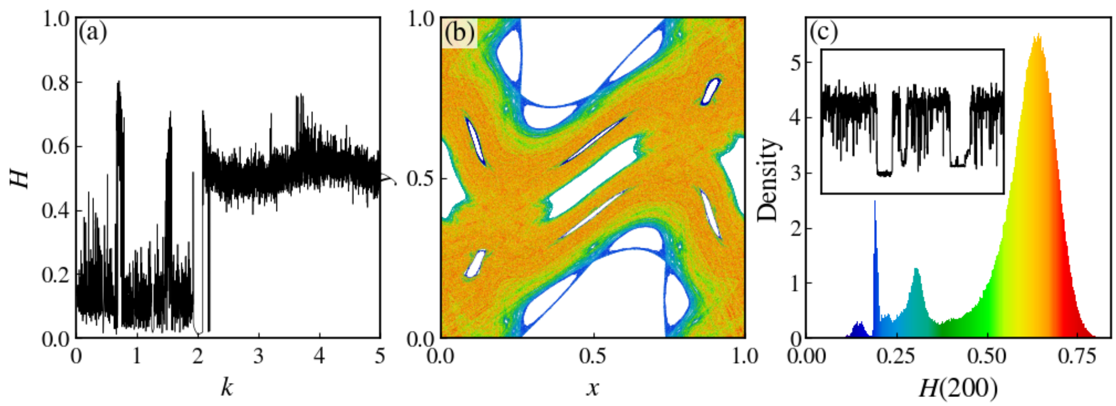
```
CPU times: user 2.72 s, sys: 76 ms, total: 2.79 s
Wall time: 2.78 s
```

[93]:
```
fontsize = 18
ps = PlotStyler(fontsize=fontsize, ticks_on_all_sides=False)
ps.apply_style()
fig, ax = plt.subplots(1, 3, figsize=(10, 3.5))
[ps.set_tick_padding(ax[i], pad_x=6) for i in range(ax.shape[0])]

ax[0].plot(k, (HE[:, 0] + HE[:, 1]) / 2, "k", lw=0.5)
ax[0].set_xlim(k.min(), k.max())
```

```python
ax[0].set_xticks([0, 1, 2, 3, 4, 5])
ax[0].set_ylim(0, 1)
ax[0].set_xlabel(r"$k$")
ax[0].set_ylabel(r"$H$")

for i in range(points.shape[0]):
    ax[1].scatter(ts[i, :, 0], ts[i, :, 1], c=ftHE_avg[i] * np.ones(finite_time), cmap="nipy_spectral", s=0.05, edgecolors="none", vmin=0, vmax=ftHE_avg.max())

ax[1].set_xlim(0, 1)
ax[1].set_ylim(0, 1)
ax[1].set_xlabel(r"$x$")
ax[1].set_ylabel(r"$y$")
ax[1].set_xticks([0, 0.5, 1])
ax[1].set_yticks([0, 0.5, 1])

counts, bins, patches = ax[2].hist(ftHE_avg, bins="auto", edgecolor='none', density=True)
# Compute bin centers
bin_centers = 0.5 * (bins[:-1] + bins[1:])

# Normalize bin centers for colormap
norm = plt.Normalize(0, bin_centers.max())
colormap = cm.nipy_spectral  # You can choose any colormap you like

# Apply color based on bin center (x position)
for center, patch in zip(bin_centers, patches):
    color = colormap(norm(center))
    patch.set_facecolor(color)

ax[2].set_xlim(bins[0], bins[-1])
ax[2].set_xlabel(r"$H(200)$")
ax[2].set_ylabel("Density")
ax[2].set_xlim(0, ftHE_avg.max())

ax_ins = ax[2].inset_axes([0.05, 0.45, 0.6, 0.45])  # [left, bottom, width, height]
ii = np.arange(ftHE_avg.shape[0]) / 1e3
ax_ins.plot(ii, ftHE_avg, "k", lw=0.75)
ax_ins.set_xlim(1, 2)
ax_ins.set_ylim(0, 1)
ax_ins.set_yticks([])
ax_ins.set_xticks([])

xbox = 0.0095
ybox = 0.9318
```

```
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}
for i in range(3):
    ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", bbox=bbox,␣
 ↪transform=ax[i].transAxes)
plt.subplots_adjust(left=0.059, bottom=0.15, right=0.995, top=0.975, wspace=0.
 ↪2, hspace=0.2)
plt.savefig("fig10.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```




# 3 Periodic orbits and manifolds

[3]:
```
ds = dds(model="standard map")
```

## 3.1 Period 1

The period-1 orbits can be found analytically. They are $(x, y) = (0, 0)$ and $(x, y) = (0.5, 0.0)$. Let us check their stability.

[7]:
```
u = [0, 0]
period = 1
stability = ds.classify_stability(u, k, period)
stability["classification"], stability["eigenvalues"]
```

[7]:
```
('saddle', array([3.18614066+0.j, 0.31385934+0.j]))
```

[8]:
```
u = [0.5, 0]
period = 1
stability = ds.classify_stability(u, k, period)
stability["classification"], stability["eigenvalues"]
```

[8]:
```
('elliptic (quasi-periodic)', array([0.25-0.96824584j, 0.25+0.96824584j]))
```

For the manifolds

[9]:
```
saddle = [0, 0]
n_points = 50000
iter_time = 12
```

[10]:
```
%%time
wu_period1 = ds.manifold(saddle, k, period, n_points=n_points,␣
 ↪iter_time=iter_time, stability="unstable")
ws_period1 = ds.manifold(saddle, k, period, n_points=n_points,␣
 ↪iter_time=iter_time, stability="stable")
```

```
CPU times: user 668 ms, sys: 32.3 ms, total: 700 ms
Wall time: 591 ms
```

### 3.2 Period 2

Now for the period-2 orbit. We know that

[12]:
```
periodic_orbit_center_p2 = [0.5, 0.5]
period = 2
stability = ds.classify_stability(periodic_orbit_center_p2, k, period)
stability["classification"], stability["eigenvalues"]
```

[12]:
```
('elliptic (quasi-periodic)', array([-0.125+0.99215674j, -0.125-0.99215674j]))
```

[13]:
```
x_range = (0.1, 0.3)
y_range = (0.3, 0.55)

period = 2
grid_size = 1000
tolerance = 2 / grid_size
x = np.linspace(x_range[0], x_range[1], grid_size)
y = np.linspace(y_range[0], y_range[1], grid_size)
X, Y = np.meshgrid(x, y)
grid_points = np.empty((grid_size, grid_size, 2), dtype=np.float64)
grid_points[:, :, 0] = X
grid_points[:, :, 1] = Y
```

[16]:
```
%%time
periodic_orbit_saddle_p2 = ds.find_periodic_orbit(grid_points, k, period,␣
 ↪tolerance=tolerance, verbose=True, tolerance_decay_factor=0.5)
```

```
Iter 0: Δorbit=[0.19398951 0.38794242], Δbounds=[0.0006046  0.00075475],
tol=2.00e-03
Iter 1: Δorbit=[6.97569804e-07 1.17822834e-05], Δbounds=[0.0013954  0.00124525],
tol=1.00e-03
Iter 2: Δorbit=[1.30339304e-05 1.63665133e-07], Δbounds=[0.0002613  0.00024525],
tol=5.00e-04
```

```
Iter 3: Δorbit=[4.37991944e-06 4.97719348e-06], Δbounds=[0.0002387  0.00025475],
tol=2.50e-04
Iter 4: Δorbit=[2.51783269e-06 3.64189359e-06], Δbounds=[1.13033454e-05
4.75475475e-06], tol=1.25e-04
Iter 5: Δorbit=[5.20590344e-07 7.58901280e-06], Δbounds=[0.0001137  0.00012025],
tol=6.25e-05
Iter 6: Δorbit=[2.05217904e-06 9.98578301e-06], Δbounds=[1.85330511e-05
2.07930027e-05], tol=3.13e-05
Iter 7: Δorbit=[5.93479191e-08 4.74338684e-08], Δbounds=[1.27169489e-05
1.04569973e-05], tol=1.56e-05
Iter 8: Δorbit=[5.55501459e-09 4.63855188e-09], Δbounds=[2.90805114e-06
5.16800271e-06], tol=7.81e-06
Iter 9: Δorbit=[1.13948853e-08 1.26565990e-08], Δbounds=[4.90444886e-06
2.65484364e-06], tol=3.91e-06
Iter 10: Δorbit=[1.20663627e-08 9.67360858e-10], Δbounds=[1.43970534e-07
1.25140636e-06], tol=1.95e-06
Iter 11: Δorbit=[1.75831010e-08 2.26119669e-08], Δbounds=[1.80915447e-06
1.07375837e-06], tol=9.77e-07
Iter 12: Δorbit=[2.02341387e-08 1.29956005e-08], Δbounds=[1.98754065e-07
9.71958682e-08], tol=4.88e-07
Iter 13: Δorbit=[5.86458457e-09 1.11955997e-08], Δbounds=[2.89527185e-07
3.91085382e-07], tol=2.44e-07
Iter 14: Δorbit=[9.33819166e-09 1.48005910e-08], Δbounds=[4.53865599e-08
6.16028015e-08], tol=1.22e-07
Iter 15: Δorbit=[5.59878710e-09 3.51646018e-09], Δbounds=[7.66837527e-08
6.04675110e-08], tol=6.10e-08
Iter 16: Δorbit=[5.21837212e-09 3.12760384e-09], Δbounds=[7.20493995e-09
5.67645264e-10], tol=3.05e-08
Iter 17: Δorbit=[9.95678762e-10 3.13003140e-09], Δbounds=[2.33126382e-08
2.99499329e-08], tol=1.53e-08
Iter 18: Δorbit=[3.29292796e-09 5.37633266e-09], Δbounds=[2.73324702e-09
3.77845560e-09], tol=7.63e-09
Iter 19: Δorbit=[6.32768810e-10 3.99263123e-10], Δbounds=[4.89614749e-09
3.85093896e-09], tol=3.81e-09
Iter 20: Δorbit=[5.62221686e-10 3.48825469e-10], Δbounds=[2.13964624e-10
9.25814980e-12], tol=1.91e-09
Iter 21: Δorbit=[1.28868194e-10 3.35720673e-10], Δbounds=[1.69338399e-09
1.89809046e-09], tol=9.54e-10
Iter 22: Δorbit=[3.75867615e-10 5.57253244e-10], Δbounds=[5.1507576e-11
1.7791324e-12], tol=4.77e-10
Iter 23: Δorbit=[3.14295534e-11 8.41683945e-11], Δbounds=[4.25329605e-10
4.75057993e-10], tol=2.38e-10
Iter 24: Δorbit=[9.41929590e-11 1.40366163e-10], Δbounds=[1.40543688e-11
1.12737597e-12], tol=1.19e-10
Iter 25: Δorbit=[8.37774294e-12 2.14752660e-11], Δbounds=[1.05999043e-10
1.18081933e-10], tol=5.96e-11
Iter 26: Δorbit=[2.36753395e-11 3.52891050e-11], Δbounds=[3.20959925e-12
1.21380683e-12], tol=2.98e-11
```

```
Iter 27: Δorbit=[1.98430161e-12 5.39102096e-12], Δbounds=[2.65927280e-11
2.85884649e-11], tol=1.49e-11
Iter 28: Δorbit=[5.86569682e-12 8.86546392e-12], Δbounds=[8.84126106e-13
1.07913678e-12], tol=7.45e-12
Iter 29: Δorbit=[3.45362627e-13 1.20320420e-12], Δbounds=[6.63460953e-12
6.86078971e-12], tol=3.73e-12
Iter 30: Δorbit=[1.40709666e-12 2.18519647e-12], Δbounds=[1.99673611e-13
6.00464123e-13], tol=1.86e-12
Iter 31: Δorbit=[5.34572386e-14 1.66144876e-13], Δbounds=[1.66294756e-12
1.71174186e-12], tol=9.31e-13
Iter 32: Δorbit=[3.48970852e-13 5.42954570e-13], Δbounds=[5.24857935e-14
1.46604950e-13], tol=4.66e-13
Iter 33: Δorbit=[1.15463195e-14 4.32986980e-14], Δbounds=[4.14834833e-13
4.28934666e-13], tol=2.33e-13
Iter 34: Δorbit=[8.74578188e-14 1.36335387e-13], Δbounds=[1.34336986e-14
3.65818487e-14], tol=1.16e-13
Iter 35: Δorbit=[2.80331314e-15 1.09356968e-14], Δbounds=[1.03500541e-13
1.07136522e-13], tol=5.82e-14
Iter 36: Δorbit=[2.18713936e-14 3.40838469e-14], Δbounds=[3.21964677e-15
9.15933995e-15], tol=2.91e-14
Iter 37: Δorbit=[7.49400542e-16 2.72004641e-15], Δbounds=[2.59514632e-14
2.68118860e-14], tol=1.46e-14
Iter 38: Δorbit=[5.41233725e-15 8.60422844e-15], Δbounds=[9.15933995e-16
2.27595720e-15], tol=7.28e-15
Iter 39: Δorbit=[2.22044605e-16 8.88178420e-16], Δbounds=[6.46704912e-15
6.77236045e-15], tol=3.64e-15
Iter 40: Δorbit=[1.38777878e-15 2.27595720e-15], Δbounds=[1.11022302e-16
7.21644966e-16], tol=1.82e-15
Iter 41: Δorbit=[1.11022302e-16 3.88578059e-16], Δbounds=[1.72084569e-15
1.49880108e-15], tol=9.09e-16
Iter 42: Δorbit=[3.05311332e-16 6.66133815e-16], Δbounds=[1.38777878e-16
1.11022302e-16], tol=4.55e-16
Converged after 42 iterations
CPU times: user 6.48 s, sys: 83 ms, total: 6.56 s
Wall time: 6.6 s
```

[18]:
```
periodic_orbit_saddle_p2, ds.classify_stability(periodic_orbit_saddle_p2, k,␣
 ↪period)
```

[18]:
```
(array([0.19397649, 0.38795298]),
 {'classification': 'saddle',
  'eigenvalues': array([4.09176343+0.j, 0.24439341+0.j]),
  'eigenvectors': array([[ 0.89240544+0.j, -0.69908845+0.j],
         [ 0.45123445+0.j,  0.7150352 +0.j]])})
```

[28]:
```
%%time
n_points = 50000
iter_time = 17
```

```
wu_period2 = ds.manifold(periodic_orbit_saddle_p2, k, period,␣
 ↪n_points=n_points, iter_time=iter_time, stability="unstable")
ws_period2 = ds.manifold(periodic_orbit_saddle_p2, k, period,␣
 ↪n_points=n_points, iter_time=iter_time, stability="stable")
```

```
CPU times: user 237 ms, sys: 9.48 ms, total: 246 ms
Wall time: 43.2 ms
```

### 3.3 Period 3

There two period 3 period orbits. Let us find the lower one first.

#### 3.3.1 Lower period 3

For the center.

[33]:
```
# Define the symmetry line
symmetry_line = lambda v, parameters: 0.5 * np.ones_like(v)
# Define the type of the function, i.e., x = g(y)
axis = 1
# Define the period
period = 3
# Define the range of the initial search
y_range = (0.2, 0.4)
# Define the number of points in the range
num_points = 10000
# Define the initial conditions
points = np.linspace(y_range[0], y_range[1], num_points)
tolerance = 2 / num_points
```

[34]:
```
%%time
periodic_orbit_center_period3_lower = ds.find_periodic_orbit(points, k, period,␣
 ↪tolerance=tolerance, symmetry_line=symmetry_line, axis=axis, verbose=True,␣
 ↪tolerance_decay_factor=0.8)
```

```
Iter 0: Δorbit=[0.5        0.38569857], Δbounds=[0.0004  0.00036], tol=2.00e-04
Iter 1: Δorbit=[0.00000000e+00 1.60214241e-06], Δbounds=[0.00032    0.00028364],
tol=1.60e-04
Iter 2: Δorbit=[0. 0.], Δbounds=[0.000256   0.00022692], tol=1.28e-04
Iter 3: Δorbit=[0.00000000e+00 1.13473509e-08], Δbounds=[0.0002048  0.00018152],
tol=1.02e-04
Iter 4: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[0.00016384 0.00014523],
tol=8.19e-05
Iter 5: Δorbit=[0. 0.], Δbounds=[0.00013107 0.00011618], tol=6.55e-05
Iter 6: Δorbit=[0. 0.], Δbounds=[1.0485760e-04 9.2947534e-05], tol=5.24e-05
Iter 7: Δorbit=[0. 0.], Δbounds=[8.3886080e-05 7.4358005e-05], tol=4.19e-05
Iter 8: Δorbit=[0. 0.], Δbounds=[6.71088640e-05 5.94864062e-05], tol=3.36e-05
Iter 9: Δorbit=[0. 0.], Δbounds=[5.36870912e-05 4.75891248e-05], tol=2.68e-05
Iter 10: Δorbit=[0. 0.], Δbounds=[4.29496730e-05 3.80712998e-05], tol=2.15e-05
```

```
Iter 11: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[3.43597384e-05
3.04570399e-05], tol=1.72e-05
Iter 12: Δorbit=[0.00000000e+00 1.52300433e-09], Δbounds=[2.74877907e-05
2.43625859e-05], tol=1.37e-05
Iter 13: Δorbit=[0. 0.], Δbounds=[2.19902326e-05 1.94928178e-05], tol=1.10e-05
Iter 14: Δorbit=[0. 0.], Δbounds=[1.75921860e-05 1.55939724e-05], tol=8.80e-06
Iter 15: Δorbit=[0. 0.], Δbounds=[1.40737488e-05 1.24752068e-05], tol=7.04e-06
Iter 16: Δorbit=[0. 0.], Δbounds=[1.12589991e-05 9.98016249e-06], tol=5.63e-06
Iter 17: Δorbit=[0. 0.], Δbounds=[9.00719925e-06 7.98413029e-06], tol=4.50e-06
Iter 18: Δorbit=[0. 0.], Δbounds=[7.2057594e-06 6.3873042e-06], tol=3.60e-06
Iter 19: Δorbit=[0. 0.], Δbounds=[5.76460752e-06 5.10984337e-06], tol=2.88e-06
Iter 20: Δorbit=[0. 0.], Δbounds=[4.61168602e-06 4.08787469e-06], tol=2.31e-06
Iter 21: Δorbit=[0. 0.], Δbounds=[3.68934881e-06 3.27029975e-06], tol=1.84e-06
Iter 22: Δorbit=[0. 0.], Δbounds=[2.95147905e-06 2.61623980e-06], tol=1.48e-06
Iter 23: Δorbit=[0. 0.], Δbounds=[2.36118324e-06 2.09299184e-06], tol=1.18e-06
Iter 24: Δorbit=[0. 0.], Δbounds=[1.88894659e-06 1.67439347e-06], tol=9.44e-07
Iter 25: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.51115727e-06
1.33951478e-06], tol=7.56e-07
Iter 26: Δorbit=[0.00000000e+00 6.69824196e-11], Δbounds=[1.20892582e-06
1.07147786e-06], tol=6.04e-07
Iter 27: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[9.67140656e-07
8.57303191e-07], tol=4.84e-07
Iter 28: Δorbit=[0. 0.], Δbounds=[7.73712525e-07 6.85830159e-07], tol=3.87e-07
Iter 29: Δorbit=[0. 0.], Δbounds=[6.18970020e-07 5.48665398e-07], tol=3.09e-07
Iter 30: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[4.95176016e-07
4.38932188e-07], tol=2.48e-07
Iter 31: Δorbit=[0. 0.], Δbounds=[3.96140813e-07 3.51145764e-07], tol=1.98e-07
Iter 32: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[3.1691265e-07
2.8091661e-07], tol=1.58e-07
Iter 33: Δorbit=[0. 0.], Δbounds=[2.53530120e-07 2.24733288e-07], tol=1.27e-07
Iter 34: Δorbit=[0.00000000e+00 1.12378995e-11], Δbounds=[2.02824096e-07
1.79764155e-07], tol=1.01e-07
Iter 35: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.62259277e-07
1.43831608e-07], tol=8.11e-08
Iter 36: Δorbit=[0. 0.], Δbounds=[1.29807421e-07 1.15063207e-07], tol=6.49e-08
Iter 37: Δorbit=[0. 0.], Δbounds=[1.03845937e-07 9.20507789e-08], tol=5.19e-08
Iter 38: Δorbit=[0. 0.], Δbounds=[8.30767497e-08 7.36406013e-08], tol=4.15e-08
Iter 39: Δorbit=[0. 0.], Δbounds=[6.64613998e-08 5.89124833e-08], tol=3.32e-08
Iter 40: Δorbit=[0. 0.], Δbounds=[5.31691198e-08 4.71299865e-08], tol=2.66e-08
Iter 41: Δorbit=[0.00000000e+00 2.35672593e-12], Δbounds=[4.25352959e-08
3.76992756e-08], tol=2.13e-08
Iter 42: Δorbit=[0. 0.], Δbounds=[3.40282367e-08 3.01636744e-08], tol=1.70e-08
Iter 43: Δorbit=[0. 0.], Δbounds=[2.72225893e-08 2.41305036e-08], tol=1.36e-08
Iter 44: Δorbit=[0. 0.], Δbounds=[2.17780715e-08 1.93044475e-08], tol=1.09e-08
Iter 45: Δorbit=[0. 0.], Δbounds=[1.74224572e-08 1.54435534e-08], tol=8.71e-09
Iter 46: Δorbit=[0. 0.], Δbounds=[1.39379657e-08 1.23548433e-08], tol=6.97e-09
Iter 47: Δorbit=[0. 0.], Δbounds=[1.11503726e-08 9.88387450e-09], tol=5.58e-09
Iter 48: Δorbit=[0. 0.], Δbounds=[8.92029806e-09 7.90709964e-09], tol=4.46e-09
```

```
Iter 49: Δorbit=[0. 0.], Δbounds=[7.13623849e-09 6.32567976e-09], tol=3.57e-09
Iter 50: Δorbit=[0. 0.], Δbounds=[5.70899078e-09 5.06054365e-09], tol=2.85e-09
Iter 51: Δorbit=[0. 0.], Δbounds=[4.56719262e-09 4.04843503e-09], tol=2.28e-09
Iter 52: Δorbit=[0. 0.], Δbounds=[3.65375413e-09 3.23874794e-09], tol=1.83e-09
Iter 53: Δorbit=[0. 0.], Δbounds=[2.92300323e-09 2.59099842e-09], tol=1.46e-09
Iter 54: Δorbit=[0. 0.], Δbounds=[2.33840258e-09 2.07279871e-09], tol=1.17e-09
Iter 55: Δorbit=[0.00000000e+00 1.03583808e-13], Δbounds=[1.87072208e-09
1.65803171e-09], tol=9.35e-10
Iter 56: Δorbit=[0. 0.], Δbounds=[1.49657770e-09 1.32661238e-09], tol=7.48e-10
Iter 57: Δorbit=[0. 0.], Δbounds=[1.19726212e-09 1.06127085e-09], tol=5.99e-10
Iter 58: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[9.57809665e-10
8.49018633e-10], tol=4.79e-10
Iter 59: Δorbit=[0. 0.], Δbounds=[7.66247732e-10 6.79214573e-10], tol=3.83e-10
Iter 60: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[6.12998263e-10
5.43371792e-10], tol=3.06e-10
Iter 61: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[4.90398611e-10
4.34697389e-10], tol=2.45e-10
Iter 62: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[3.92318844e-10
3.47757934e-10], tol=1.96e-10
Iter 63: Δorbit=[0. 0.], Δbounds=[3.13855109e-10 2.78206347e-10], tol=1.57e-10
Iter 64: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.51084042e-10
2.22565077e-10], tol=1.26e-10
Iter 65: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.00867267e-10
1.78052018e-10], tol=1.00e-10
Iter 66: Δorbit=[0. 0.], Δbounds=[1.60693847e-10 1.42441614e-10], tol=8.03e-11
Iter 67: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.28555000e-10
1.13953291e-10], tol=6.43e-11
Iter 68: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.02844011e-10
9.11626330e-11], tol=5.14e-11
Iter 69: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[8.22752422e-11
7.29301064e-11], tol=4.11e-11
Iter 70: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[6.58201826e-11
5.83439963e-11], tol=3.29e-11
Iter 71: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[5.26561017e-11
4.66752192e-11], tol=2.63e-11
Iter 72: Δorbit=[0.00000000e+00 2.33146835e-15], Δbounds=[4.21249702e-11
3.73355791e-11], tol=2.11e-11
Iter 73: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[3.36999317e-11
2.98726599e-11], tol=1.68e-11
Iter 74: Δorbit=[0. 0.], Δbounds=[2.69599898e-11 2.38977726e-11], tol=1.35e-11
Iter 75: Δorbit=[0. 0.], Δbounds=[2.15679141e-11 1.91181515e-11], tol=1.08e-11
Iter 76: Δorbit=[0. 0.], Δbounds=[1.72544201e-11 1.52946544e-11], tol=8.63e-12
Iter 77: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.38034584e-11
1.22356569e-11], tol=6.90e-12
Iter 78: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.10427778e-11
9.78850334e-12], tol=5.52e-12
Iter 79: Δorbit=[0.00000000e+00 5.55111512e-16], Δbounds=[8.83426665e-12
7.82984788e-12], tol=4.42e-12
```

```
Iter 80: Δorbit=[0. 0.], Δbounds=[7.06740222e-12 6.26465546e-12], tol=3.53e-12
Iter 81: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[5.65392178e-12
5.01176878e-12], tol=2.83e-12
Iter 82: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[4.52310411e-12
4.00934841e-12], tol=2.26e-12
Iter 83: Δorbit=[0. 0.], Δbounds=[3.61849439e-12 3.20754534e-12], tol=1.81e-12
Iter 84: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.89479551e-12
2.56566990e-12], tol=1.45e-12
Iter 85: Δorbit=[0. 0.], Δbounds=[2.31586972e-12 2.05285788e-12], tol=1.16e-12
Iter 86: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.85268467e-12
1.64240843e-12], tol=9.26e-13
Iter 87: Δorbit=[0.00000000e+00 1.66533454e-16], Δbounds=[1.48214774e-12
1.31378242e-12], tol=7.41e-13
Iter 88: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.18571819e-12
1.05104814e-12], tol=5.93e-13
Iter 89: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[9.48574552e-13
8.40771897e-13], tol=4.74e-13
Iter 90: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[7.58892948e-13
6.72684131e-13], tol=3.79e-13
Iter 91: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[6.07069950e-13
5.38236122e-13], tol=3.04e-13
Iter 92: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[4.85667062e-13
4.30544489e-13], tol=2.43e-13
Iter 93: Δorbit=[0. 0.], Δbounds=[3.88578059e-13 3.44502205e-13], tol=1.94e-13
Iter 94: Δorbit=[0. 0.], Δbounds=[3.10862447e-13 2.75612866e-13], tol=1.55e-13
Iter 95: Δorbit=[0. 0.], Δbounds=[2.48689958e-13 2.20490293e-13], tol=1.24e-13
Iter 96: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.98951966e-13
1.76358927e-13], tol=9.95e-14
Iter 97: Δorbit=[0. 0.], Δbounds=[1.59150471e-13 1.40998324e-13], tol=7.96e-14
Iter 98: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.27287070e-13
1.12965193e-13], tol=6.37e-14
Iter 99: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.01862963e-13
9.03166431e-14], tol=5.09e-14
Iter 100: Δorbit=[0. 0.], Δbounds=[8.14903700e-14 7.22755189e-14], tol=4.07e-14
Iter 101: Δorbit=[0. 0.], Δbounds=[6.52256027e-14 5.78981307e-14], tol=3.26e-14
Iter 102: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[5.21804822e-14
4.64073224e-14], tol=2.61e-14
Iter 103: Δorbit=[0. 0.], Δbounds=[4.17443857e-14 3.69704267e-14], tol=2.09e-14
Iter 104: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[3.33622019e-14
2.97539771e-14], tol=1.67e-14
Iter 105: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.66453526e-14
2.35922393e-14], tol=1.33e-14
Iter 106: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.13162821e-14
1.89293026e-14], tol=1.07e-14
Iter 107: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.70974346e-14
1.52100554e-14], tol=8.54e-15
Iter 108: Δorbit=[0. 0.], Δbounds=[1.37112544e-14 1.21569421e-14], tol=6.84e-15
Iter 109: Δorbit=[0. 0.], Δbounds=[1.09356968e-14 9.76996262e-15], tol=5.47e-15
```

```
Iter 110: Δorbit=[0. 0.], Δbounds=[8.71525074e-15 7.88258347e-15], tol=4.37e-15
Iter 111: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[7.04991621e-15
6.16173779e-15], tol=3.50e-15
Iter 112: Δorbit=[0. 0.], Δbounds=[5.55111512e-15 5.10702591e-15], tol=2.80e-15
Iter 113: Δorbit=[0. 0.], Δbounds=[4.44089210e-15 3.99680289e-15], tol=2.24e-15
Iter 114: Δorbit=[0. 0.], Δbounds=[3.55271368e-15 3.16413562e-15], tol=1.79e-15
Iter 115: Δorbit=[0. 0.], Δbounds=[2.88657986e-15 2.55351296e-15], tol=1.43e-15
Iter 116: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[2.2759572e-15
2.0539126e-15], tol=1.15e-15
Iter 117: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.83186799e-15
1.60982339e-15], tol=9.17e-16
Iter 118: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.49880108e-15
1.27675648e-15], tol=7.34e-16
Iter 119: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.16573418e-15
1.11022302e-15], tol=5.87e-16
Iter 120: Δorbit=[0.00000000e+00 1.66533454e-16], Δbounds=[8.88178420e-16
8.32667268e-16], tol=4.70e-16
Converged at iteration 120
CPU times: user 307 ms, sys: 8.05 ms, total: 315 ms
Wall time: 317 ms
```

[35]:
```python
periodic_orbit_center_period3_lower, ds.
↪classify_stability(periodic_orbit_center_period3_lower, k, period)
```

[35]:
```
(array([0.5       , 0.38569696]),
 {'classification': 'elliptic (quasi-periodic)',
  'eigenvalues': array([-0.93105758-0.36487228j, -0.93105758+0.36487228j]),
  'eigenvectors': array([[-0.31198762+0.06967997j, -0.31198762-0.06967997j],
         [ 0.94752753+0.j        ,  0.94752753+0.j        ]])})
```

For the saddle

[36]:
```python
# Define the symmetry line
symmetry_line = lambda v, parameters: 0.0 * np.ones_like(v)
# Define the type of the function, i.e., x = g(y)
axis = 1
# Define the period
period = 3
# Define the range of the initial search
y_range = (0.2, 0.3)
# Define the number of points in the range
num_points = 10000
# Define the initial conditions
points = np.linspace(y_range[0], y_range[1], num_points)
tolerance = 2 / num_points
```

[37]:
```python
%%time
```

```
periodic_orbit_saddle_period3_lower = ds.find_periodic_orbit(points, k, period,␣
↪tolerance=tolerance, symmetry_line=symmetry_line, axis=axis, verbose=True,␣
↪tolerance_decay_factor=0.8)
```

```
Iter 0: Δorbit=[0.         0.25381538], Δbounds=[0.0004     0.00033999],
tol=2.00e-04
Iter 1: Δorbit=[0.00000000e+00 9.11275046e-06], Δbounds=[0.00032    0.00026407],
tol=1.60e-04
Iter 2: Δorbit=[0.00000000e+00 5.61195081e-06], Δbounds=[0.000256   0.00021126],
tol=1.28e-04
Iter 3: Δorbit=[0.00000000e+00 4.46865611e-06], Δbounds=[0.0002048  0.00016901],
tol=1.02e-04
Iter 4: Δorbit=[0.00000000e+00 3.57488714e-06], Δbounds=[0.00016384 0.00013521],
tol=8.19e-05
Iter 5: Δorbit=[0.00000000e+00 2.86667714e-06], Δbounds=[0.00013107 0.00010815],
tol=6.55e-05
Iter 6: Δorbit=[0.00000000e+00 2.29305393e-06], Δbounds=[1.04857600e-04
8.65239849e-05], tol=5.24e-05
Iter 7: Δorbit=[0.00000000e+00 1.83449193e-06], Δbounds=[8.38860800e-05
6.92187978e-05], tol=4.19e-05
Iter 8: Δorbit=[0.00000000e+00 1.46758527e-06], Δbounds=[6.71088640e-05
5.53751044e-05], tol=3.36e-05
Iter 9: Δorbit=[0.00000000e+00 1.17130059e-06], Δbounds=[5.36870912e-05
4.43056104e-05], tol=2.68e-05
Iter 10: Δorbit=[0.00000000e+00 9.41588379e-07], Δbounds=[4.2949673e-05
3.5443552e-05], tol=2.15e-05
Iter 11: Δorbit=[0.00000000e+00 7.49706094e-07], Δbounds=[3.43597384e-05
2.83550002e-05], tol=1.72e-05
Iter 12: Δorbit=[0.00000000e+00 5.99768231e-07], Δbounds=[2.74877907e-05
2.26839733e-05], tol=1.37e-05
Iter 13: Δorbit=[0.0000000e+00 4.8208264e-07], Δbounds=[2.19902326e-05
1.81471832e-05], tol=1.10e-05
Iter 14: Δorbit=[0.00000000e+00 3.83851309e-07], Δbounds=[1.75921860e-05
1.45177458e-05], tol=8.80e-06
Iter 15: Δorbit=[0.00000000e+00 3.07806991e-07], Δbounds=[1.40737488e-05
1.16127448e-05], tol=7.04e-06
Iter 16: Δorbit=[0.00000000e+00 2.46214812e-07], Δbounds=[1.12589991e-05
9.29044196e-06], tol=5.63e-06
Iter 17: Δorbit=[0.00000000e+00 1.96977067e-07], Δbounds=[9.00719925e-06
7.43231185e-06], tol=4.50e-06
Iter 18: Δorbit=[0.00000000e+00 1.57580769e-07], Δbounds=[7.20575940e-06
5.94585655e-06], tol=3.60e-06
Iter 19: Δorbit=[0.00000000e+00 1.26064765e-07], Δbounds=[5.76460752e-06
4.75668404e-06], tol=2.88e-06
Iter 20: Δorbit=[0.00000000e+00 1.00613929e-07], Δbounds=[4.61168602e-06
3.80582315e-06], tol=2.31e-06
Iter 21: Δorbit=[0.00000000e+00 8.08818302e-08], Δbounds=[3.68934881e-06
3.04457790e-06], tol=1.84e-06
```

```
Iter 22: Δorbit=[0.00000000e+00 6.43992624e-08], Δbounds=[2.95147905e-06
2.43567598e-06], tol=1.48e-06
Iter 23: Δorbit=[0.00000000e+00 5.16414949e-08], Δbounds=[2.36118324e-06
1.94829487e-06], tol=1.18e-06
Iter 24: Δorbit=[0.00000000e+00 4.13079821e-08], Δbounds=[1.88894659e-06
1.55867758e-06], tol=9.44e-07
Iter 25: Δorbit=[0.00000000e+00 3.30472695e-08], Δbounds=[1.51115727e-06
1.24693500e-06], tol=7.56e-07
Iter 26: Δorbit=[0.00000000e+00 2.63753128e-08], Δbounds=[1.20892582e-06
9.97673905e-07], tol=6.04e-07
Iter 27: Δorbit=[0.00000000e+00 2.12026908e-08], Δbounds=[9.67140656e-07
7.98117794e-07], tol=4.84e-07
Iter 28: Δorbit=[0.00000000e+00 1.68818795e-08], Δbounds=[7.73712525e-07
6.38497849e-07], tol=3.87e-07
Iter 29: Δorbit=[0.00000000e+00 1.35375082e-08], Δbounds=[6.18970020e-07
5.10733811e-07], tol=3.09e-07
Iter 30: Δorbit=[0.00000000e+00 1.08286397e-08], Δbounds=[4.95176016e-07
4.08597977e-07], tol=2.48e-07
Iter 31: Δorbit=[0.00000000e+00 8.64271149e-09], Δbounds=[3.96140813e-07
3.26917393e-07], tol=1.98e-07
Iter 32: Δorbit=[0.00000000e+00 6.94768937e-09], Δbounds=[3.16912650e-07
2.61527305e-07], tol=1.58e-07
Iter 33: Δorbit=[0.0000000e+00 5.5318557e-09], Δbounds=[2.53530120e-07
2.09222964e-07], tol=1.27e-07
Iter 34: Δorbit=[0.00000000e+00 4.43597042e-09], Δbounds=[2.02824096e-07
1.67357257e-07], tol=1.01e-07
Iter 35: Δorbit=[0.00000000e+00 3.54832869e-09], Δbounds=[1.62259277e-07
1.33889385e-07], tol=8.11e-08
Iter 36: Δorbit=[0.00000000e+00 2.83204371e-09], Δbounds=[1.29807421e-07
1.07124291e-07], tol=6.49e-08
Iter 37: Δorbit=[0.00000000e+00 2.27126207e-09], Δbounds=[1.03845937e-07
8.56865539e-08], tol=5.19e-08
Iter 38: Δorbit=[0.00000000e+00 1.81673665e-09], Δbounds=[8.30767497e-08
6.85514263e-08], tol=4.15e-08
Iter 39: Δorbit=[0.00000000e+00 1.45343554e-09], Δbounds=[6.64613998e-08
5.48407710e-08], tol=3.32e-08
Iter 40: Δorbit=[0.00000000e+00 1.16274063e-09], Δbounds=[5.31691198e-08
4.38726795e-08], tol=2.66e-08
Iter 41: Δorbit=[0.00000000e+00 9.30193811e-10], Δbounds=[4.25352959e-08
3.50981330e-08], tol=2.13e-08
Iter 42: Δorbit=[0.00000000e+00 7.44154849e-10], Δbounds=[3.40282367e-08
2.80785082e-08], tol=1.70e-08
Iter 43: Δorbit=[0.00000000e+00 5.93919858e-10], Δbounds=[2.72225894e-08
2.24656144e-08], tol=1.36e-08
Iter 44: Δorbit=[0.0000000e+00 4.7744203e-10], Δbounds=[2.17780715e-08
1.79720158e-08], tol=1.09e-08
Iter 45: Δorbit=[0.00000000e+00 3.80146192e-10], Δbounds=[1.74224572e-08
1.43776932e-08], tol=8.71e-09
```

```
Iter 46: Δorbit=[0.00000000e+00 3.04837544e-10], Δbounds=[1.39379657e-08
1.15007031e-08], tol=6.97e-09
Iter 47: Δorbit=[0.00000000e+00 2.43839282e-10], Δbounds=[1.11503726e-08
9.20080850e-09], tol=5.58e-09
Iter 48: Δorbit=[0.00000000e+00 1.95076677e-10], Δbounds=[8.92029808e-09
7.36060501e-09], tol=4.46e-09
Iter 49: Δorbit=[0.00000000e+00 1.55692348e-10], Δbounds=[7.13623846e-09
5.88922727e-09], tol=3.57e-09
Iter 50: Δorbit=[0.00000000e+00 1.25158606e-10], Δbounds=[5.70899077e-09
4.71125589e-09], tol=2.85e-09
Iter 51: Δorbit=[0.00000000e+00 9.96530636e-11], Δbounds=[4.56719262e-09
3.76902609e-09], tol=2.28e-09
Iter 52: Δorbit=[0.00000000e+00 7.99112998e-11], Δbounds=[3.65375409e-09
3.01484027e-09], tol=1.83e-09
Iter 53: Δorbit=[0.00000000e+00 6.39210351e-11], Δbounds=[2.92300327e-09
2.41193671e-09], tol=1.46e-09
Iter 54: Δorbit=[0.00000000e+00 5.11382048e-11], Δbounds=[2.33840262e-09
1.92953842e-09], tol=1.17e-09
Iter 55: Δorbit=[0.00000000e+00 4.08137968e-11], Δbounds=[1.87072210e-09
1.54382568e-09], tol=9.35e-10
Iter 56: Δorbit=[0.00000000e+00 3.28096439e-11], Δbounds=[1.49657768e-09
1.23502741e-09], tol=7.48e-10
Iter 57: Δorbit=[0.00000000e+00 2.61233812e-11], Δbounds=[1.19726214e-09
9.88027604e-10], tol=5.99e-10
Iter 58: Δorbit=[0.00000000e+00 2.09483542e-11], Δbounds=[9.57809713e-10
7.90322308e-10], tol=4.79e-10
Iter 59: Δorbit=[0.00000000e+00 1.67564851e-11], Δbounds=[7.66247770e-10
6.32274677e-10], tol=3.83e-10
Iter 60: Δorbit=[0.00000000e+00 1.33739686e-11], Δbounds=[6.12998216e-10
5.05880171e-10], tol=3.06e-10
Iter 61: Δorbit=[0.00000000e+00 1.07510112e-11], Δbounds=[4.90398573e-10
4.04693945e-10], tol=2.45e-10
Iter 62: Δorbit=[0.00000000e+00 8.56009708e-12], Δbounds=[3.92318858e-10
3.23756855e-10], tol=1.96e-10
Iter 63: Δorbit=[0.00000000e+00 6.86434243e-12], Δbounds=[3.13855087e-10
2.58972788e-10], tol=1.57e-10
Iter 64: Δorbit=[0.0000000e+00 5.4907745e-12], Δbounds=[2.51084069e-10
2.07183770e-10], tol=1.26e-10
Iter 65: Δorbit=[0.00000000e+00 4.39276393e-12], Δbounds=[2.00867256e-10
1.65746084e-10], tol=1.00e-10
Iter 66: Δorbit=[0.00000000e+00 3.50586227e-12], Δbounds=[1.60693804e-10
1.32613587e-10], tol=8.03e-11
Iter 67: Δorbit=[0.00000000e+00 2.81835666e-12], Δbounds=[1.28555044e-10
1.06088083e-10], tol=6.43e-11
Iter 68: Δorbit=[0.00000000e+00 2.24392727e-12], Δbounds=[1.02844035e-10
8.48709436e-11], tol=5.14e-11
Iter 69: Δorbit=[0.00000000e+00 1.79944948e-12], Δbounds=[8.22752279e-11
6.78880840e-11], tol=4.11e-11
```

```
Iter 70: Δorbit=[0.00000000e+00 1.43934864e-12], Δbounds=[6.58201823e-11
5.43119438e-11], tol=3.29e-11
Iter 71: Δorbit=[0.00000000e+00 1.15157883e-12], Δbounds=[5.26561458e-11
4.34492997e-11], tol=2.63e-11
Iter 72: Δorbit=[0.00000000e+00 9.21207555e-13], Δbounds=[4.21249167e-11
3.47595841e-11], tol=2.11e-11
Iter 73: Δorbit=[0.00000000e+00 7.35189687e-13], Δbounds=[3.36999333e-11
2.78110313e-11], tol=1.68e-11
Iter 74: Δorbit=[0.0000000e+00 5.8969496e-13], Δbounds=[2.69599467e-11
2.22455387e-11], tol=1.35e-11
Iter 75: Δorbit=[0.00000000e+00 4.71622741e-13], Δbounds=[2.15679573e-11
1.77969306e-11], tol=1.08e-11
Iter 76: Δorbit=[0.00000000e+00 3.77364806e-13], Δbounds=[1.72543659e-11
1.42375001e-11], tol=8.63e-12
Iter 77: Δorbit=[0.0000000e+00 3.0186964e-13], Δbounds=[1.38034927e-11
1.13900001e-11], tol=6.90e-12
Iter 78: Δorbit=[0.00000000e+00 2.41473508e-13], Δbounds=[1.10427942e-11
9.11204445e-12], tol=5.52e-12
Iter 79: Δorbit=[0.00000000e+00 1.92734717e-13], Δbounds=[8.83423532e-12
7.29050154e-12], tol=4.42e-12
Iter 80: Δorbit=[0.00000000e+00 1.54598556e-13], Δbounds=[7.06738826e-12
5.83150195e-12], tol=3.53e-12
Iter 81: Δorbit=[0.00000000e+00 1.23623334e-13], Δbounds=[5.65391061e-12
4.66537919e-12], tol=2.83e-12
Iter 82: Δorbit=[0.00000000e+00 9.89208715e-14], Δbounds=[4.52312849e-12
3.73229225e-12], tol=2.26e-12
Iter 83: Δorbit=[0.00000000e+00 7.91033905e-14], Δbounds=[3.61850279e-12
2.98583380e-12], tol=1.81e-12
Iter 84: Δorbit=[0.00000000e+00 6.33382236e-14], Δbounds=[2.89480223e-12
2.38864484e-12], tol=1.45e-12
Iter 85: Δorbit=[0.00000000e+00 5.05706588e-14], Δbounds=[2.31584178e-12
1.91091587e-12], tol=1.16e-12
Iter 86: Δorbit=[0.00000000e+00 4.06341627e-14], Δbounds=[1.85267343e-12
1.52888813e-12], tol=9.26e-13
Iter 87: Δorbit=[0.00000000e+00 3.23630012e-14], Δbounds=[1.48213874e-12
1.22313271e-12], tol=7.41e-13
Iter 88: Δorbit=[0.00000000e+00 2.58681965e-14], Δbounds=[1.18571099e-12
9.78495063e-13], tol=5.93e-13
Iter 89: Δorbit=[0.00000000e+00 2.08166817e-14], Δbounds=[9.48568795e-13
7.82818255e-13], tol=4.74e-13
Iter 90: Δorbit=[0.00000000e+00 1.65978342e-14], Δbounds=[7.58855036e-13
6.26221297e-13], tol=3.79e-13
Iter 91: Δorbit=[0.0000000e+00 1.3211654e-14], Δbounds=[6.07084029e-13
5.00877118e-13], tol=3.04e-13
Iter 92: Δorbit=[0.0000000e+00 1.0658141e-14], Δbounds=[4.85667223e-13
4.00846023e-13], tol=2.43e-13
Iter 93: Δorbit=[0.00000000e+00 8.49320614e-15], Δbounds=[3.88533778e-13
3.20687921e-13], tol=1.94e-13
```

```
Iter 94: Δorbit=[0.0000000e+00 6.7168493e-15], Δbounds=[3.10827023e-13
2.56517030e-13], tol=1.55e-13
Iter 95: Δorbit=[0.00000000e+00 5.49560397e-15], Δbounds=[2.48661618e-13
2.05280237e-13], tol=1.24e-13
Iter 96: Δorbit=[0.0000000e+00 4.3298698e-15], Δbounds=[1.98929295e-13
1.64257496e-13], tol=9.95e-14
Iter 97: Δorbit=[0.00000000e+00 3.44169138e-15], Δbounds=[1.59143436e-13
1.31228362e-13], tol=7.96e-14
Iter 98: Δorbit=[0.00000000e+00 2.83106871e-15], Δbounds=[1.27314749e-13
1.05138120e-13], tol=6.37e-14
Iter 99: Δorbit=[0.00000000e+00 2.22044605e-15], Δbounds=[1.01851799e-13
8.39883718e-14], tol=5.09e-14
Iter 100: Δorbit=[0.00000000e+00 1.77635684e-15], Δbounds=[8.14814391e-14
6.72795153e-14], tol=4.07e-14
Iter 101: Δorbit=[0.00000000e+00 1.38777878e-15], Δbounds=[6.51851512e-14
5.37903055e-14], tol=3.26e-14
Iter 102: Δorbit=[0.00000000e+00 1.11022302e-15], Δbounds=[5.21481210e-14
4.30766534e-14], tol=2.61e-14
Iter 103: Δorbit=[0.00000000e+00 9.99200722e-16], Δbounds=[4.17184968e-14
3.45279361e-14], tol=2.09e-14
Iter 104: Δorbit=[0.00000000e+00 6.66133815e-16], Δbounds=[3.33747974e-14
2.75890422e-14], tol=1.67e-14
Iter 105: Δorbit=[0.00000000e+00 5.55111512e-16], Δbounds=[2.66998379e-14
2.19269047e-14], tol=1.33e-14
Iter 106: Δorbit=[0.00000000e+00 5.55111512e-16], Δbounds=[2.13598704e-14
1.75970349e-14], tol=1.07e-14
Iter 107: Δorbit=[0.00000000e+00 3.33066907e-16], Δbounds=[1.70878963e-14
1.40998324e-14], tol=8.54e-15
Iter 108: Δorbit=[0.00000000e+00 3.33066907e-16], Δbounds=[1.36703170e-14
1.13242749e-14], tol=6.84e-15
Iter 109: Δorbit=[0.00000000e+00 2.77555756e-16], Δbounds=[1.09362536e-14
9.15933995e-15], tol=5.47e-15
Iter 110: Δorbit=[0.00000000e+00 1.66533454e-16], Δbounds=[8.74900290e-15
7.32747196e-15], tol=4.37e-15
Iter 111: Δorbit=[0.00000000e+00 1.66533454e-16], Δbounds=[6.99920232e-15
5.88418203e-15], tol=3.50e-15
Iter 112: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[5.59936186e-15
4.60742555e-15], tol=2.80e-15
Iter 113: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[4.47948948e-15
3.66373598e-15], tol=2.24e-15
Iter 114: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[3.58359159e-15
2.88657986e-15], tol=1.79e-15
Iter 115: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[2.86687327e-15
2.38697950e-15], tol=1.43e-15
Iter 116: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[2.29349862e-15
1.94289029e-15], tol=1.15e-15
Iter 117: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.83479889e-15
1.55431223e-15], tol=9.17e-16
```

```
Iter 118: Δorbit=[0.00000000e+00 5.55111512e-17], Δbounds=[1.46783911e-15
1.22124533e-15], tol=7.34e-16
Iter 119: Δorbit=[0. 0.], Δbounds=[1.17427129e-15 1.05471187e-15], tol=5.87e-16
Iter 120: Δorbit=[0. 0.], Δbounds=[9.39417033e-16 7.21644966e-16], tol=4.70e-16
Converged at iteration 120
CPU times: user 250 ms, sys: 9.02 ms, total: 259 ms
Wall time: 274 ms
```

[38]:
```
periodic_orbit_saddle_period3_lower, ds.
↪classify_stability(periodic_orbit_saddle_period3_lower, k, period)
```

[38]:
```
(array([0.        , 0.25377828]),
 {'classification': 'saddle',
  'eigenvalues': array([5.90789859+0.j, 0.16926492+0.j]),
  'eigenvectors': array([[ 0.84347661+0.j,  0.94680784+0.j],
         [ 0.53716591+0.j, -0.32179949+0.j]])})
```

### 3.3.2 Upper period 3

For the center

[39]:
```
# Define the symmetry line
symmetry_line = lambda v, parameters: 0.5 * np.ones_like(v)
# Define the type of the function, i.e., x = g(y)
axis = 1
# Define the period
period = 3
# Define the range of the initial search
y_range = (0.55, 0.65)
# Define the number of points in the range
num_points = 10000
# Define the initial conditions
points = np.linspace(y_range[0], y_range[1], num_points)
tolerance = 2 / num_points
```

[40]:
```
%%time
periodic_orbit_center_period3_upper = ds.find_periodic_orbit(points, k, period,␣
↪tolerance=tolerance, symmetry_line=symmetry_line, axis=axis, verbose=True,␣
↪tolerance_decay_factor=0.7)
```

```
Iter 0: Δorbit=[0.5        0.61430143], Δbounds=[0.0004  0.00037], tol=2.00e-04
Iter 1: Δorbit=[0.00000000e+00 1.60964791e-06], Δbounds=[0.00028    0.00024818],
tol=1.40e-04
Iter 2: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[0.000196   0.00017374],
tol=9.80e-05
Iter 3: Δorbit=[0. 0.], Δbounds=[0.0001372  0.00012161], tol=6.86e-05
Iter 4: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[9.60400000e-05
8.51301066e-05], tol=4.80e-05
Iter 5: Δorbit=[0. 0.], Δbounds=[6.72280000e-05 5.95910657e-05], tol=3.36e-05
```

```
Iter 6: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[4.70596000e-05
4.17137468e-05], tol=2.35e-05
Iter 7: Δorbit=[0.0000000e+00 4.1717918e-09], Δbounds=[3.29417200e-05
2.91996227e-05], tol=1.65e-05
Iter 8: Δorbit=[0.00000000e+00 2.92025426e-09], Δbounds=[2.30592040e-05
2.04397359e-05], tol=1.15e-05
Iter 9: Δorbit=[0.00000000e+00 2.04417816e-09], Δbounds=[1.61414428e-05
1.43078151e-05], tol=8.07e-06
Iter 10: Δorbit=[0.00000000e+00 1.43092482e-09], Δbounds=[1.12990100e-05
1.00154706e-05], tol=5.65e-06
Iter 11: Δorbit=[0.00000000e+00 5.00823605e-10], Δbounds=[7.90930697e-06
7.01183106e-06], tol=3.95e-06
Iter 12: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[5.53651488e-06
4.90749073e-06], tol=2.77e-06
Iter 13: Δorbit=[0. 0.], Δbounds=[3.87556042e-06 3.43531447e-06], tol=1.94e-06
Iter 14: Δorbit=[0. 0.], Δbounds=[2.71289229e-06 2.40471377e-06], tol=1.36e-06
Iter 15: Δorbit=[0. 0.], Δbounds=[1.89902460e-06 1.68330021e-06], tol=9.50e-07
Iter 16: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.32931722e-06
1.17831009e-06], tol=6.65e-07
Iter 17: Δorbit=[0. 0.], Δbounds=[9.30522056e-07 8.24817070e-07], tol=4.65e-07
Iter 18: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[6.51365439e-07
5.77371949e-07], tol=3.26e-07
Iter 19: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[4.55955807e-07
4.04160364e-07], tol=2.28e-07
Iter 20: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[3.19169065e-07
2.82912255e-07], tol=1.60e-07
Iter 21: Δorbit=[0. 0.], Δbounds=[2.23418346e-07 1.98038578e-07], tol=1.12e-07
Iter 22: Δorbit=[0.00000000e+00 1.98058236e-11], Δbounds=[1.56392842e-07
1.38627005e-07], tol=7.82e-08
Iter 23: Δorbit=[0.00000000e+00 1.38642431e-11], Δbounds=[1.09474989e-07
9.70389034e-08], tol=5.47e-08
Iter 24: Δorbit=[0.00000000e+00 9.70490355e-12], Δbounds=[7.66324926e-08
6.79272324e-08], tol=3.83e-08
Iter 25: Δorbit=[0.00000000e+00 6.79334367e-12], Δbounds=[5.36427447e-08
4.75490626e-08], tol=2.68e-08
Iter 26: Δorbit=[0.00000000e+00 2.37765363e-12], Δbounds=[3.75499214e-08
3.32890993e-08], tol=1.88e-08
Iter 27: Δorbit=[0. 0.], Δbounds=[2.62849449e-08 2.32986141e-08], tol=1.31e-08
Iter 28: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.83994615e-08
1.63093667e-08], tol=9.20e-09
Iter 29: Δorbit=[0. 0.], Δbounds=[1.28796230e-08 1.14165264e-08], tol=6.44e-09
Iter 30: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[9.01573616e-09
7.99157140e-09], tol=4.51e-09
Iter 31: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[6.31101527e-09
5.59409963e-09], tol=3.16e-09
Iter 32: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[4.41771064e-09
3.91586963e-09], tol=2.21e-09
Iter 33: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[3.09239745e-09
```

```
2.74110890e-09], tol=1.55e-09
Iter 34: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.16467821e-09
1.91877625e-09], tol=1.08e-09
Iter 35: Δorbit=[0. 0.], Δbounds=[1.51527479e-09 1.34314337e-09], tol=7.58e-10
Iter 36: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.06069237e-09
9.40200362e-10], tol=5.30e-10
Iter 37: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[7.42484685e-10
6.58140431e-10], tol=3.71e-10
Iter 38: Δorbit=[0. 0.], Δbounds=[5.19739196e-10 4.60698146e-10], tol=2.60e-10
Iter 39: Δorbit=[0. 0.], Δbounds=[3.63817421e-10 3.22488702e-10], tol=1.82e-10
Iter 40: Δorbit=[0. 0.], Δbounds=[2.54672283e-10 2.25742092e-10], tol=1.27e-10
Iter 41: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.78270509e-10
1.58019375e-10], tol=8.91e-11
Iter 42: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.24789346e-10
1.10613518e-10], tol=6.24e-11
Iter 43: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[8.73525696e-11
7.74296183e-11], tol=4.37e-11
Iter 44: Δorbit=[0. 0.], Δbounds=[6.11468098e-11 5.42006440e-11], tol=3.06e-11
Iter 45: Δorbit=[0. 0.], Δbounds=[4.28028168e-11 3.79405396e-11], tol=2.14e-11
Iter 46: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.99619773e-11
2.65585332e-11], tol=1.50e-11
Iter 47: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.09733897e-11
1.85909066e-11], tol=1.05e-11
Iter 48: Δorbit=[0.00000000e+00 2.22044605e-16], Δbounds=[1.46813672e-11
1.30135902e-11], tol=7.34e-12
Iter 49: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.02768904e-11
9.10937992e-12], tol=5.14e-12
Iter 50: Δorbit=[0. 0.], Δbounds=[7.19385662e-12 6.37667696e-12], tol=3.60e-12
Iter 51: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[5.03574960e-12
4.46376269e-12], tol=2.52e-12
Iter 52: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[3.52495810e-12
3.12461168e-12], tol=1.76e-12
Iter 53: Δorbit=[0.00000000e+00 2.22044605e-16], Δbounds=[2.46752618e-12
2.18702834e-12], tol=1.23e-12
Iter 54: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.72728498e-12
1.53110857e-12], tol=8.64e-13
Iter 55: Δorbit=[0. 0.], Δbounds=[1.20903287e-12 1.07158726e-12], tol=6.05e-13
Iter 56: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[8.46378523e-13
7.50399742e-13], tol=4.23e-13
Iter 57: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[5.92415006e-13
5.25024468e-13], tol=2.96e-13
Iter 58: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[4.14723811e-13
3.67816888e-13], tol=2.07e-13
Iter 59: Δorbit=[0. 0.], Δbounds=[2.90267810e-13 2.57460719e-13], tol=1.45e-13
Iter 60: Δorbit=[0. 0.], Δbounds=[2.03170814e-13 1.80189197e-13], tol=1.02e-13
Iter 61: Δorbit=[0.00000000e+00 2.22044605e-16], Δbounds=[1.42275081e-13
1.26343380e-13], tol=7.11e-14
Iter 62: Δorbit=[0.00000000e+00 2.22044605e-16], Δbounds=[9.95314942e-14
```

```
8.81517082e-14], tol=4.98e-14
Iter 63: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[6.97220059e-14
6.18394225e-14], tol=3.49e-14
Iter 64: Δorbit=[0. 0.], Δbounds=[4.87943019e-14 4.35207426e-14], tol=2.44e-14
Iter 65: Δorbit=[0. 0.], Δbounds=[3.41948692e-14 3.05311332e-14], tol=1.71e-14
Iter 66: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.39253062e-14
2.14273044e-14], tol=1.20e-14
Iter 67: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.67088565e-14
1.48769885e-14], tol=8.37e-15
Iter 68: Δorbit=[0.00000000e+00 2.22044605e-16], Δbounds=[1.17683641e-14
1.05471187e-14], tol=5.86e-15
Iter 69: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[8.21565038e-15
7.43849426e-15], tol=4.10e-15
Iter 70: Δorbit=[0. 0.], Δbounds=[5.77315973e-15 5.21804822e-15], tol=2.87e-15
Iter 71: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[3.99680289e-15
3.66373598e-15], tol=2.01e-15
Iter 72: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.83106871e-15
2.66453526e-15], tol=1.41e-15
Iter 73: Δorbit=[0. 0.], Δbounds=[1.99840144e-15 1.88737914e-15], tol=9.84e-16
Iter 74: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.33226763e-15
1.33226763e-15], tol=6.89e-16
Iter 75: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[9.43689571e-16
8.88178420e-16], tol=4.82e-16
Converged at iteration 75
CPU times: user 166 ms, sys: 8.14 ms, total: 174 ms
Wall time: 193 ms
```

[41]:
```python
periodic_orbit_center_period3_upper, ds.
 ↪classify_stability(periodic_orbit_center_period3_upper, k, period)
```

[41]:
```
(array([0.5       , 0.61430304]),
 {'classification': 'elliptic (quasi-periodic)',
  'eigenvalues': array([-0.93105758+0.36487228j, -0.93105758-0.36487228j]),
  'eigenvectors': array([[-0.31198762-0.06967997j, -0.31198762+0.06967997j],
         [ 0.94752753+0.j        ,  0.94752753+0.j        ]])})
```

Now for the saddle

[42]:
```python
# Define the symmetry line
symmetry_line = lambda v, parameters: 0.0 * np.ones_like(v)
# Define the type of the function, i.e., x = g(y)
axis = 1
# Define the period
period = 3
# Define the range of the initial search
y_range = (0.65, 0.8)
# Define the number of points in the range
num_points = 10000
```

```python
# Define the initial conditions
points = np.linspace(y_range[0], y_range[1], num_points)
tolerance = 2 / num_points
```

[43]:
```python
%%time
periodic_orbit_saddle_period3_upper = ds.find_periodic_orbit(points, k, period,
    tolerance=tolerance, symmetry_line=symmetry_line, axis=axis, verbose=True,
    tolerance_decay_factor=0.7)
```

```
Iter 0: Δorbit=[0.         0.74625713], Δbounds=[0.0004   0.000355],
tol=2.00e-04
Iter 1: Δorbit=[0.00000000e+00 1.09172033e-05], Δbounds=[0.00028    0.00023108],
tol=1.40e-04
Iter 2: Δorbit=[0.00000000e+00 7.34897484e-06], Δbounds=[0.000196   0.00016173],
tol=9.80e-05
Iter 3: Δorbit=[0.00000000e+00 5.14346138e-06], Δbounds=[0.0001372  0.00011321],
tol=6.86e-05
Iter 4: Δorbit=[0.00000000e+00 3.60054473e-06], Δbounds=[9.60400000e-05
7.92487804e-05], tol=4.80e-05
Iter 5: Δorbit=[0.00000000e+00 2.52036325e-06], Δbounds=[6.72280000e-05
5.54742305e-05], tol=3.36e-05
Iter 6: Δorbit=[0.00000000e+00 1.76425696e-06], Δbounds=[4.70596000e-05
3.88319489e-05], tol=2.35e-05
Iter 7: Δorbit=[0.00000000e+00 1.23497947e-06], Δbounds=[3.2941720e-05
2.7182366e-05], tol=1.65e-05
Iter 8: Δorbit=[0.00000000e+00 8.64485689e-07], Δbounds=[2.3059204e-05
1.9027656e-05], tol=1.15e-05
Iter 9: Δorbit=[0.00000000e+00 6.04188496e-07], Δbounds=[1.61414428e-05
1.33212622e-05], tol=8.07e-06
Iter 10: Δorbit=[0.00000000e+00 4.23658503e-07], Δbounds=[1.12990100e-05
9.32326921e-06], tol=5.65e-06
Iter 11: Δorbit=[0.00000000e+00 2.96509612e-07], Δbounds=[7.90930697e-06
6.52652787e-06], tol=3.95e-06
Iter 12: Δorbit=[0.00000000e+00 2.07564343e-07], Δbounds=[5.53651488e-06
4.56853400e-06], tol=2.77e-06
Iter 13: Δorbit=[0.00000000e+00 1.45065461e-07], Δbounds=[3.87556042e-06
3.19843597e-06], tol=1.94e-06
Iter 14: Δorbit=[0.00000000e+00 1.01720436e-07], Δbounds=[2.71289229e-06
2.23851680e-06], tol=1.36e-06
Iter 15: Δorbit=[0.00000000e+00 7.11919534e-08], Δbounds=[1.89902460e-06
1.56701936e-06], tol=9.50e-07
Iter 16: Δorbit=[0.00000000e+00 4.98361994e-08], Δbounds=[1.32931722e-06
1.09690501e-06], tol=6.65e-07
Iter 17: Δorbit=[0.00000000e+00 3.48850676e-08], Δbounds=[9.30522056e-07
7.67834774e-07], tol=4.65e-07
Iter 18: Δorbit=[0.00000000e+00 2.43811923e-08], Δbounds=[6.51365439e-07
5.37560945e-07], tol=3.26e-07
Iter 19: Δorbit=[0.00000000e+00 1.70961477e-08], Δbounds=[4.55955807e-07
```

```
3.76227546e-07], tol=2.28e-07
Iter 20: Δorbit=[0.00000000e+00 1.19652325e-08], Δbounds=[3.19169065e-07
2.63368940e-07], tol=1.60e-07
Iter 21: Δorbit=[0.00000000e+00 8.37597003e-09], Δbounds=[2.23418346e-07
1.84356826e-07], tol=1.12e-07
Iter 22: Δorbit=[0.00000000e+00 5.86313331e-09], Δbounds=[1.56392842e-07
1.29049991e-07], tol=7.82e-08
Iter 23: Δorbit=[0.00000000e+00 4.10420009e-09], Δbounds=[1.09474989e-07
9.03349617e-08], tol=5.47e-08
Iter 24: Δorbit=[0.00000000e+00 2.86842194e-09], Δbounds=[7.66324926e-08
6.32435123e-08], tol=3.83e-08
Iter 25: Δorbit=[0.00000000e+00 2.01134476e-09], Δbounds=[5.36427448e-08
4.42627939e-08], tol=2.68e-08
Iter 26: Δorbit=[0.00000000e+00 1.40769763e-09], Δbounds=[3.75499214e-08
3.09850925e-08], tol=1.88e-08
Iter 27: Δorbit=[0.00000000e+00 9.83875204e-10], Δbounds=[2.62849449e-08
2.16924949e-08], tol=1.31e-08
Iter 28: Δorbit=[0.00000000e+00 6.90974944e-10], Δbounds=[1.83994615e-08
1.51843123e-08], tol=9.20e-09
Iter 29: Δorbit=[0.00000000e+00 4.82150209e-10], Δbounds=[1.28796230e-08
1.06290828e-08], tol=6.44e-09
Iter 30: Δorbit=[0.00000000e+00 3.37507133e-10], Δbounds=[9.01573612e-09
7.44034845e-09], tol=4.51e-09
Iter 31: Δorbit=[0.00000000e+00 2.36626718e-10], Δbounds=[6.31101528e-09
5.20750121e-09], tol=3.16e-09
Iter 32: Δorbit=[0.00000000e+00 1.65614966e-10], Δbounds=[4.41771070e-09
3.64536101e-09], tol=2.21e-09
Iter 33: Δorbit=[0.00000000e+00 1.15934151e-10], Δbounds=[3.09239749e-09
2.55173638e-09], tol=1.55e-09
Iter 34: Δorbit=[0.00000000e+00 8.11531953e-11], Δbounds=[2.16467824e-09
1.78621784e-09], tol=1.08e-09
Iter 35: Δorbit=[0.00000000e+00 5.68074476e-11], Δbounds=[1.51527477e-09
1.25035216e-09], tol=7.58e-10
Iter 36: Δorbit=[0.00000000e+00 3.97026856e-11], Δbounds=[1.06069234e-09
8.75371664e-10], tol=5.30e-10
Iter 37: Δorbit=[0.00000000e+00 2.78397305e-11], Δbounds=[7.42484637e-10
6.12654150e-10], tol=3.71e-10
Iter 38: Δorbit=[0.0000000e+00 1.9484192e-11], Δbounds=[5.19739246e-10
4.28873492e-10], tol=2.60e-10
Iter 39: Δorbit=[0.00000000e+00 1.36181066e-11], Δbounds=[3.63817472e-10
3.00251934e-10], tol=1.82e-10
Iter 40: Δorbit=[0.00000000e+00 9.56412727e-12], Δbounds=[2.54672230e-10
2.10170548e-10], tol=1.27e-10
Iter 41: Δorbit=[0.0000000e+00 6.6735506e-12], Δbounds=[1.78270561e-10
1.47120094e-10], tol=8.91e-11
Iter 42: Δorbit=[0.00000000e+00 4.67148542e-12], Δbounds=[1.24789393e-10
1.02983955e-10], tol=6.24e-11
Iter 43: Δorbit=[0.00000000e+00 3.28015393e-12], Δbounds=[8.73525750e-11
```

```
7.20888904e-11], tol=4.37e-11
Iter 44: Δorbit=[0.00000000e+00 2.28916885e-12], Δbounds=[6.11468025e-11
5.04621900e-11], tol=3.06e-11
Iter 45: Δorbit=[0.00000000e+00 1.60238489e-12], Δbounds=[4.28027618e-11
3.53236329e-11], tol=2.14e-11
Iter 46: Δorbit=[0.00000000e+00 1.12532206e-12], Δbounds=[2.99619332e-11
2.47265541e-11], tol=1.50e-11
Iter 47: Δorbit=[0.00000000e+00 7.85038701e-13], Δbounds=[2.09733533e-11
1.73085990e-11], tol=1.05e-11
Iter 48: Δorbit=[0.00000000e+00 5.50559598e-13], Δbounds=[1.46813473e-11
1.21143096e-11], tol=7.34e-12
Iter 49: Δorbit=[0.00000000e+00 3.85136367e-13], Δbounds=[1.02769431e-11
8.48010551e-12], tol=5.14e-12
Iter 50: Δorbit=[0.00000000e+00 2.69229083e-13], Δbounds=[7.19386017e-12
5.93691762e-12], tol=3.60e-12
Iter 51: Δorbit=[0.00000000e+00 1.88959959e-13], Δbounds=[5.03570212e-12
4.15523171e-12], tol=2.52e-12
Iter 52: Δorbit=[0.00000000e+00 1.32005518e-13], Δbounds=[3.52499148e-12
2.90867330e-12], tol=1.76e-12
Iter 53: Δorbit=[0.00000000e+00 9.22595333e-14], Δbounds=[2.46749404e-12
2.03648209e-12], tol=1.23e-12
Iter 54: Δorbit=[0.00000000e+00 6.49480469e-14], Δbounds=[1.72724583e-12
1.42530432e-12], tol=8.64e-13
Iter 55: Δorbit=[0.00000000e+00 4.52970994e-14], Δbounds=[1.20907208e-12
9.97535388e-13], tol=6.05e-13
Iter 56: Δorbit=[0.00000000e+00 3.18634008e-14], Δbounds=[8.46350455e-13
6.98441305e-13], tol=4.23e-13
Iter 57: Δorbit=[0.00000000e+00 2.20934382e-14], Δbounds=[5.92445319e-13
4.88831198e-13], tol=2.96e-13
Iter 58: Δorbit=[0.00000000e+00 1.55431223e-14], Δbounds=[4.14711723e-13
3.42392781e-13], tol=2.07e-13
Iter 59: Δorbit=[0.00000000e+00 1.09912079e-14], Δbounds=[2.90298206e-13
2.39475106e-13], tol=1.45e-13
Iter 60: Δorbit=[0.00000000e+00 7.54951657e-15], Δbounds=[2.03208744e-13
1.67754699e-13], tol=1.02e-13
Iter 61: Δorbit=[0.00000000e+00 5.32907052e-15], Δbounds=[1.42246121e-13
1.17572618e-13], tol=7.11e-14
Iter 62: Δorbit=[0.00000000e+00 3.55271368e-15], Δbounds=[9.95722847e-14
8.20454815e-14], tol=4.98e-14
Iter 63: Δorbit=[0.00000000e+00 2.77555756e-15], Δbounds=[6.97005993e-14
5.76205750e-14], tol=3.49e-14
Iter 64: Δorbit=[0.00000000e+00 1.77635684e-15], Δbounds=[4.87904195e-14
4.04121181e-14], tol=2.44e-14
Iter 65: Δorbit=[0.00000000e+00 1.22124533e-15], Δbounds=[3.41532937e-14
2.81996648e-14], tol=1.71e-14
Iter 66: Δorbit=[0.00000000e+00 7.77156117e-16], Δbounds=[2.39073056e-14
1.97619698e-14], tol=1.20e-14
Iter 67: Δorbit=[0.00000000e+00 7.77156117e-16], Δbounds=[1.67351139e-14
```

```
1.37667655e-14], tol=8.37e-15
Iter 68: Δorbit=[0.00000000e+00 3.33066907e-16], Δbounds=[1.17145797e-14
9.76996262e-15], tol=5.86e-15
Iter 69: Δorbit=[0.00000000e+00 3.33066907e-16], Δbounds=[8.20020581e-15
6.77236045e-15], tol=4.10e-15
Iter 70: Δorbit=[0.00000000e+00 3.33066907e-16], Δbounds=[5.74014406e-15
4.88498131e-15], tol=2.87e-15
Iter 71: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[4.01810085e-15
3.44169138e-15], tol=2.01e-15
Iter 72: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[2.81267059e-15
2.44249065e-15], tol=1.41e-15
Iter 73: Δorbit=[0. 0.], Δbounds=[1.96886941e-15 1.66533454e-15], tol=9.84e-16
Iter 74: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[1.37820859e-15
1.22124533e-15], tol=6.89e-16
Iter 75: Δorbit=[0.00000000e+00 1.11022302e-16], Δbounds=[9.64746013e-16
7.77156117e-16], tol=4.82e-16
Converged at iteration 75
CPU times: user 167 ms, sys: 8.62 ms, total: 175 ms
Wall time: 191 ms
```

[44]:
```
periodic_orbit_saddle_period3_upper, ds.
↪classify_stability(periodic_orbit_saddle_period3_upper, k, period)
```

[44]:
```
(array([0.        , 0.74622172]),
 {'classification': 'saddle',
  'eigenvalues': array([5.90789859+0.j, 0.16926492+0.j]),
  'eigenvectors': array([[ 0.84347661+0.j,  0.94680784+0.j],
         [ 0.53716591+0.j, -0.32179949+0.j]])})
```

### 3.3.3 Now the manifolds

[70]:
```
%%time
n_points = 50000
iter_time = 18
wu_period3_lower = ds.manifold(periodic_orbit_saddle_period3_lower, k, period,␣
↪n_points=n_points, iter_time=iter_time, stability="unstable")
ws_period3_lower = ds.manifold(periodic_orbit_saddle_period3_lower, k, period,␣
↪n_points=n_points, iter_time=iter_time, stability="stable")
```

```
CPU times: user 234 ms, sys: 6.59 ms, total: 240 ms
Wall time: 39.7 ms
```

[74]:
```
%%time
n_points = 50000
iter_time = 18
wu_period3_upper = ds.manifold(periodic_orbit_saddle_period3_upper, k, period,␣
↪n_points=n_points, iter_time=iter_time, stability="unstable")
```

```
ws_period3_upper = ds.manifold(periodic_orbit_saddle_period3_upper, k, period,␣
↪n_points=n_points, iter_time=iter_time, stability="stable")
```

```
CPU times: user 252 ms, sys: 7.92 ms, total: 260 ms
Wall time: 48.9 ms
```

### 3.4 Final plot

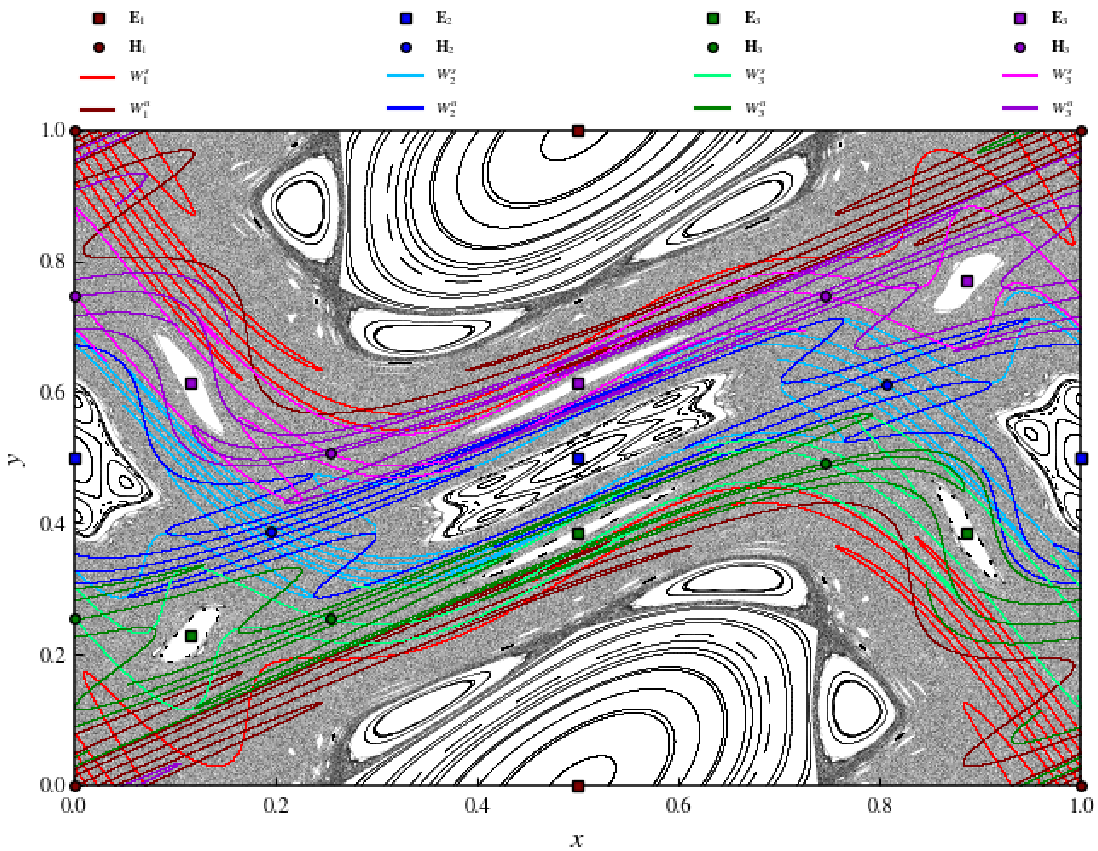
[147]:
```
num_ic = 100
total_time = 30000
np.random.seed(11331313)
u = np.random.rand(num_ic, 2)
k = 1.5
trajectories = ds.trajectory(u, k, total_time)
```

[149]:
```
plt_params(fontsize=12, legend_fontsize=7, axes_linewidth=1.1)
fig, ax = plt.subplots()
set_ticks_in(ax, pad_x=5)
plt.plot(trajectories[:, 0].T, trajectories[:, 1], "ko", markersize=0.1,␣
↪markeredgewidth=0.0)
ms = 0.75
pms = 4

plt.plot(0.5, 0, "s", markersize=pms, markeredgecolor="k", clip_on=False,␣
↪zorder=3, color="maroon", label=r"$\mathbf{E}_1$")
plt.plot(0.5, 1, "s", markersize=pms, markeredgecolor="k", clip_on=False,␣
↪zorder=3, color="maroon")
plt.plot(0, 0, "o", markersize=pms, markeredgecolor="k", clip_on=False,␣
↪zorder=3, color="maroon", label=r"$\mathbf{H}_1$")
plt.plot(0, 1, "o", markersize=pms, markeredgecolor="k", clip_on=False,␣
↪zorder=3, color="maroon")
plt.plot(1, 0, "o", markersize=pms, markeredgecolor="k", clip_on=False,␣
↪zorder=3, color="maroon")
plt.plot(1, 1, "o", markersize=pms, markeredgecolor="k", clip_on=False,␣
↪zorder=3, color="maroon")

plt.plot(0, 0, "r", label="$W^s_1$")
plt.plot(0, 0, label="$W^u_1$", color="maroon")

plt.plot(ws_period1[0][:, 0], ws_period1[0][:, 1], "o", markersize=ms,␣
↪markeredgewidth=0.0, color="red") # along v
plt.plot(ws_period1[1][:, 0], ws_period1[1][:, 1], "o", markersize=ms,␣
↪markeredgewidth=0.0, color="red") # along -v
plt.plot(wu_period1[0][:, 0], wu_period1[0][:, 1], "o", markersize=ms,␣
↪markeredgewidth=0.0, color="maroon") # along v
plt.plot(wu_period1[1][:, 0], wu_period1[1][:, 1], "o", markersize=ms,␣
↪markeredgewidth=0.0, color="maroon") # along -v
```

```
ts = ds.trajectory(periodic_orbit_center_p2, k, 2)
plt.plot(ts[:, 0], ts[:, 1], "bs", markersize=pms, markeredgecolor="k", clip_on=False, zorder=3, label=r"$\mathbf{E}_2$")
plt.plot(1, 0.5 , "bs", markersize=pms, markeredgecolor="k", clip_on=False, zorder=3)
ts = ds.trajectory(periodic_orbit_saddle_p2, k, 2)
plt.plot(ts[:, 0], ts[:, 1], "bo", markersize=pms, markeredgecolor="k", clip_on=False, zorder=3, label=r"$\mathbf{H}_2$")

plt.plot(ws_period2[0][:, 0], ws_period2[0][:, 1], "o", markersize=ms, markeredgewidth=0.0, color="deepskyblue") # along v
plt.plot(ws_period2[1][:, 0], ws_period2[1][:, 1], "o", markersize=ms, markeredgewidth=0.0, color="deepskyblue") # along -v
plt.plot(wu_period2[0][:, 0], wu_period2[0][:, 1], "bo", markersize=ms, markeredgewidth=0.0) # along v
plt.plot(wu_period2[1][:, 0], wu_period2[1][:, 1], "bo", markersize=ms, markeredgewidth=0.0) # along -v

plt.plot(0, 0, label="$W^s_2$", color="deepskyblue")
plt.plot(0, 0, label="$W^u_2$", color="blue")

ts = ds.trajectory(periodic_orbit_center_period3_lower, k, 3)
plt.plot(ts[:, 0], ts[:, 1], "gs", markersize=pms, markeredgecolor="k", clip_on=False, zorder=3, label=r"$\mathbf{E}_3$")
plt.plot(1, 0.5 , "bs", markersize=pms, markeredgecolor="k", clip_on=False, zorder=3)
ts = ds.trajectory(periodic_orbit_saddle_period3_lower, k, 3)
plt.plot(ts[:, 0], ts[:, 1], "go", markersize=pms, markeredgecolor="k", clip_on=False, zorder=3, label=r"$\mathbf{H}_3$")

plt.plot(ws_period3_lower[0][:, 0], ws_period3_lower[0][:, 1], "o", markersize=ms, markeredgewidth=0.0, color="springgreen") # along v
plt.plot(ws_period3_lower[1][:, 0], ws_period3_lower[1][:, 1], "o", markersize=ms, markeredgewidth=0.0, color="springgreen") # along -v
plt.plot(wu_period3_lower[0][:, 0], wu_period3_lower[0][:, 1], "go", markersize=ms, markeredgewidth=0.0) # along v
plt.plot(wu_period3_lower[1][:, 0], wu_period3_lower[1][:, 1], "go", markersize=ms, markeredgewidth=0.0) # along -v

plt.plot(0, 0, label="$W^s_3$", color="springgreen")
plt.plot(0, 0, label="$W^u_3$", color="green")

ts = ds.trajectory(periodic_orbit_center_period3_upper, k, 3)
plt.plot(ts[:, 0], ts[:, 1], "s", markersize=pms, markeredgecolor="k", clip_on=False, zorder=3, color="darkviolet", label=r"$\mathbf{E}_3$")
```

```
ts = ds.trajectory(periodic_orbit_saddle_period3_upper, k, 3)
plt.plot(ts[:, 0], ts[:, 1], "o", markersize=pms, markeredgecolor="k",␣
↪clip_on=False, zorder=3, color="darkviolet", label=r"$\mathbf{H}_3$")

plt.plot(ws_period3_upper[0][:, 0], ws_period3_upper[0][:, 1], "o",␣
↪markersize=ms, markeredgewidth=0.0, color="fuchsia") # along v
plt.plot(ws_period3_upper[1][:, 0], ws_period3_upper[1][:, 1], "o",␣
↪markersize=ms, markeredgewidth=0.0, color="fuchsia") # along -v
plt.plot(wu_period3_upper[0][:, 0], wu_period3_upper[0][:, 1], "o",␣
↪markersize=ms, markeredgewidth=0.0, color="darkviolet") # along v
plt.plot(wu_period3_upper[1][:, 0], wu_period3_upper[1][:, 1], "o",␣
↪markersize=ms, markeredgewidth=0.0, color="darkviolet") # along -v

plt.plot(0, 0, label="$W^s_3$", color="fuchsia")
plt.plot(0, 0, label="$W^u_3$", color="darkviolet")

# plt.legend(loc="upper center", fontsize=8, frameon=False, handlelength=1.5,␣
↪handletextpad=0.5, borderpad=0.5, bbox_to_anchor=(0.5, 1.5), ncol=4)
plt.legend(bbox_to_anchor=(0, 1.0, 1, 0.2), loc="lower left",
                mode="expand", borderaxespad=0, ncol=4, frameon=False,␣
↪fancybox=False)
plt.xlim(0, 1)
plt.ylim(0, 1)
plt.xlabel(r"$x$")
plt.ylabel(r"$y$")
plt.subplots_adjust(left=0.065, bottom=0.08, right=0.987, top=0.88, wspace=0.2,␣
↪hspace=0.2)
plt.savefig("fig11.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```




# 4 Escape

## 4.1 Survival probability

[96]:
```
ds = dds(model="leonel map")
```

[103]:
```
eps, gamma = 1e-3, 1
parameters = [eps, gamma]
```

[188]:
```
total_time = 5000000
u = [np.pi, 1e-15]
trajectory = ds.trajectory(u, total_time, parameters=parameters)
```

[97]:
```
ds.info["parameters"]
```

[97]: ['eps', 'gamma']

[116]:
```
max_time = 1000000
num_ic = 1000000
```

```
np.random.seed(13)
x_range = (0, 2 * np.pi, num_ic)
y_range = (-1e-14, 1e-14, num_ic)
x = np.random.uniform(*x_range)
y = np.random.uniform(*y_range)
y_esc = np.logspace(np.log10(1e-3), np.log10(0.025), 25)
x_esc = (0, 2 * np.pi)
sp, times = [], []
```

[117]:
```
%%time
for i in range(y_esc.shape[0]):
    exit = np.array([[x_esc[0], x_esc[1]], [-y_esc[i], y_esc[i]]])
    escape = np.array(Parallel(n_jobs=-1)(delayed(ds.escape_analysis)([x[j],␣
↪y[j]], max_time, exit, parameters=parameters, escape="exiting") for j in␣
↪range(num_ic)))
    time, survival_probability = ds.survival_probability(escape[:, 1], escape[:
↪, 1].max())
    times.append(time)
    sp.append(survival_probability)
```

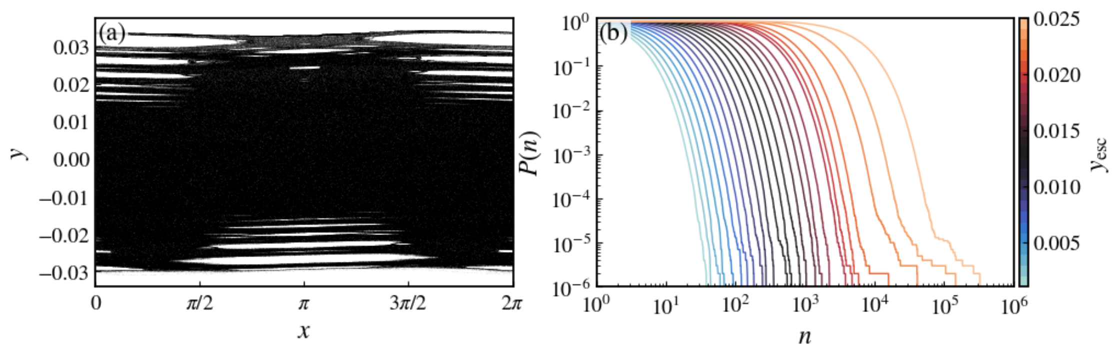
```
CPU times: user 5min 51s, sys: 5.32 s, total: 5min 56s
Wall time: 7min 54s
```

[121]:
```
colors = sns.color_palette("icefire", len(y_esc))
cmap = ListedColormap(colors)
Y_esc = np.array(y_esc)
norm = mpl.colors.Normalize(vmin=min(Y_esc), vmax=max(Y_esc))
sm = mpl.cm.ScalarMappable(cmap=cmap, norm=norm)
```

[191]:
```
fontsize=17
ps = PlotStyler(fontsize=fontsize, ticks_on_all_sides=False, markersize=0.1,␣
↪markeredgewidth=0)
ps.apply_style()

fig, ax = plt.subplots(1, 2, figsize=(10, 3))
[ps.set_tick_padding(ax[i], pad_x=5) for i in range(ax.shape[0])]
plt.subplots_adjust(left=0.075, bottom=0.16, right=1.065, top=0.975)

ax[0].plot(trajectory[:, 0], trajectory[:, 1], "ko")
ax[0].set_xlim(0, 2 * np.pi)
ax[0].set_xlabel("$x$")
ax[0].set_ylabel("$y$")
ax[0].set_xticks([0, np.pi/2, np.pi, 3 * np.pi /2, 2 * np.pi], [r"$0$", r"$\pi/
↪2$", r"$\pi$", r"$3\pi/2$", r"$2\pi$"])
ax[0].set_yticks([-0.03, -0.02, -0.01, 0, 0.01, 0.02, 0.03])

for i in range(y_esc.shape[0]):
```

```
    ax[1].plot(times[i], sp[i], color=colors[i])
ax[1].set_xscale("log")
ax[1].set_yscale("log")
ax[1].set_ylim(1 / num_ic, 1.2e0)
ax[1].set_xlim(1e0, 1e6)
ax[1].set_xlabel("$n$")
ax[1].set_ylabel("$P(n)$")
fig.colorbar(sm, ax=ax, pad=0.005, aspect=30, label=r"$y_{\mathrm{esc}}$")


xbox = 0.0066
ybox = 0.923
bbox = {"facecolor": "w", "pad": 1, "alpha": 0.75, "linewidth": 0.0}
[ax[i].text(xbox, ybox, f"({ascii_lowercase[i]})", transform=ax[i].transAxes,
 ↪bbox=bbox) for i in range(ax.shape[0])]

plt.savefig("fig12.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```




### 4.2 Escape basins

[6]:
```
from numba import njit
```

[7]:
```
@njit
def weiss_map(u, parameters):
    k = parameters[0]
    x, y = u
    y_new = y - k * np.sin(x)
    x_new = (x + k * (y_new ** 2 - 1) + np.pi) % (2 * np.pi) - np.pi

    return np.array([x_new, y_new])
```

[8]:
```
ds = dds(mapping=weiss_map, system_dimension=2, number_of_parameters=1)
```

[38]:
```
import numpy as np

centers = np.array([[0.0, -1.1],
                    [np.pi - 0.1, 1.0]], dtype=np.float64)
size_exit = 0.2
```

[39]:
```
ks = [0.5, 0.55, 0.60, 0.70]
total_time = 10000
```

[40]:
```
grid_size = 1000
x_range = (-np.pi, np.pi, grid_size)
y_range = (-np.pi, np.pi, grid_size)
X = np.linspace(*x_range)
Y = np.linspace(*y_range)
```

[41]:
```
from joblib import Parallel, delayed
import itertools
```

[42]:
```
escapes = np.zeros((len(ks), grid_size, grid_size, 2))
```

[43]:
```
%%time
for i, k in enumerate(ks):

    escape = Parallel(n_jobs=-1)(
        delayed(ds.escape_analysis)([x, y], total_time, centers, parameters=k,␣
 ↪hole_size=size_exit)
        for x, y in itertools.product(X, Y)
    )
    escape = np.array(escape).reshape(grid_size, grid_size, 2)

    escapes[i, :, :, :] = escape
```

```
CPU times: user 55.1 s, sys: 1.37 s, total: 56.4 s
Wall time: 3min 39s
```

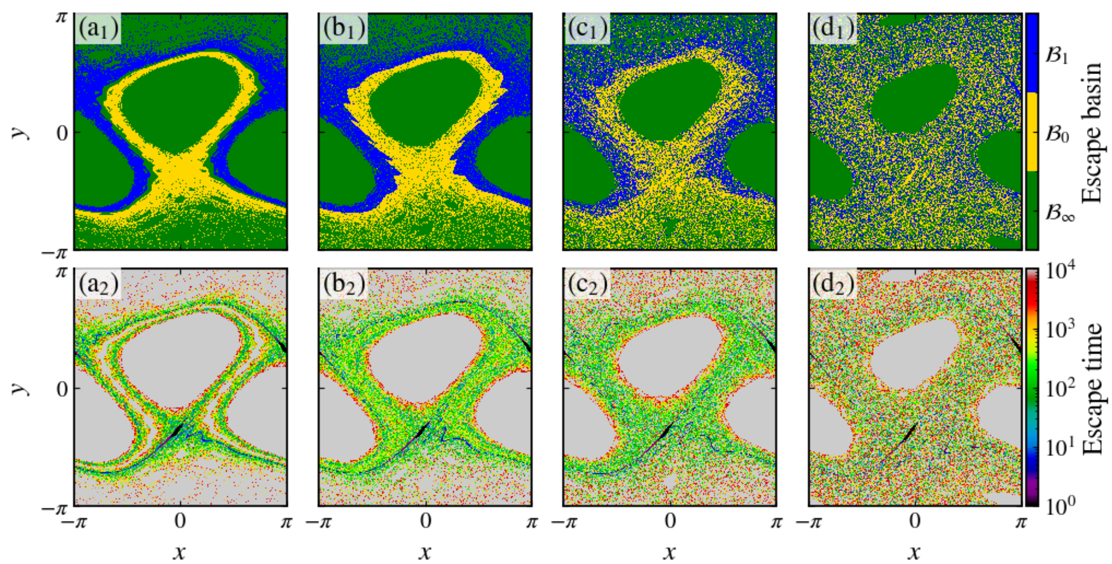
[44]:
```
from matplotlib.colors import ListedColormap, BoundaryNorm
import matplotlib as mpl
```

[45]:
```
colors = ["green", "gold", "blue"]
cmap = ListedColormap(colors)
bounds = [-1.5, -0.5, 0.5, 1.5]
norm = BoundaryNorm(boundaries=bounds, ncolors=len(colors))
```

[46]:
```
ps = PlotStyler(fontsize=18)
ps.apply_style()
fig, ax = plt.subplots(2, 4, sharex=True, sharey=True, figsize=(10, 5))
# plt.tight_layout(pad=0)
```

```
plt.subplots_adjust(left=0.055, bottom=0.095, top=0.995, right=0.945, hspace=0.
↪08, wspace=0.15)

x_grid, y_grid = np.meshgrid(X, Y, indexing='ij')
for i, k in enumerate(ks):
    hm1 = ax[0, i].pcolormesh(x_grid, y_grid, escapes[i, :, :, 0], cmap=cmap,␣
↪norm=norm)
    hm2 = ax[1, i].pcolormesh(x_grid, y_grid, escapes[i, :, :, 1],␣
↪cmap="nipy_spectral", norm=mpl.colors.LogNorm(vmin=1e0, vmax=total_time))
    ax[1, i].set_xlabel(r"$x$")

ax[0, 0].set_ylabel(r"$y$")
ax[1, 0].set_ylabel(r"$y$")
ax[0, 0].set_xticks([-np.pi, 0,  np.pi])
ax[0, 0].set_xticklabels([r"$-\pi$", r"$0$", r"$\pi$"])
ax[0, 0].set_yticks([-np.pi, 0, np.pi])
ax[0, 0].set_yticklabels([r"$-\pi$", r"$0$", r"$\pi$"])
ax[0, 0].set_xlim(-np.pi, np.pi)
ax[0, 0].set_ylim(-np.pi, np.pi)
cbar1 = fig.colorbar(hm1, ax=ax[0, :], aspect=20, pad=0.005, fraction=0.02)
cbar1.set_label(r"Escape basin")
cbar1.set_ticks([-1, 0, 1])
cbar1.set_ticklabels([r"$\mathcal{B}_\infty$", r"$\mathcal{B}_0$",␣
↪r"$\mathcal{B}_1$"])

cbar2 = fig.colorbar(hm2, ax=ax[1, :], aspect=20, pad=0.005, fraction=0.02)
cbar2.set_label(r"Escape time")

xbox = 0.0143
ybox = 0.908
bbox = {"facecolor": "w", "alpha": 0.75, "linewidth": 0.0, "pad": 1}
for i in range(4):
    ax[0, i].text(xbox, ybox, f"({ascii_lowercase[i]}$_1$)", transform=ax[0, i].
↪transAxes, bbox=bbox)
    ax[1, i].text(xbox, ybox, f"({ascii_lowercase[i]}$_2$)", transform=ax[1, i].
↪transAxes, bbox=bbox)

plt.savefig("fig13.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```




```
[63]: ks = np.linspace(0.2, 1.0, 100)
      escape_basins = np.zeros((len(ks), grid_size, grid_size))
      total_time = 10000
      Sb = []
      Sbb = []
      D = []
```

```
[64]: from pycandy import BasinMetrics
```

```
[65]: x_grid, y_grid = np.meshgrid(X, Y, indexing="ij")
```

```
[ ]: %%time
     for i, k in enumerate(ks):

         escape = Parallel(n_jobs=-1)(
             delayed(ds.escape_analysis)([x, y], total_time, centers, parameters=k,␣
      ↪hole_size=size_exit)
             for x, y in itertools.product(X, Y)
         )
         escape = np.array(escape).reshape(grid_size, grid_size, 2)
         escape_basins[i] = escape[:, :, 0]

         bm = BasinMetrics(escape[:, :, 0])
         basin_entropy = bm.basin_entropy(5, log_base=2)
         Sb.append(basin_entropy[0])
         Sbb.append(basin_entropy[1])
         eps, f = bm.uncertainty_fraction(x_grid, y_grid, )
         alpha, _ = np.polyfit(np.log(eps), np.log(f), 1)
```

```
    D.append(2 - alpha)
```

[66]:
```
import pandas as pd
```

[68]:
```
for i, k in enumerate(ks):
    df = f"escape_basin_i={i}.dat"
    df = pd.read_csv(df, header=None, sep=r"\s+")
    escape_basins[i] = np.array(df[2]).reshape(grid_size, grid_size)
```

[69]:
```
for i, k in enumerate(ks):
    bm = BasinMetrics(escape_basins[i, :, :])
    basin_entropy = bm.basin_entropy(5, log_base=2)
    Sb.append(basin_entropy[0])
    Sbb.append(basin_entropy[1])
    eps, f = bm.uncertainty_fraction(x_grid, y_grid, )
    alpha, _ = np.polyfit(np.log(eps), np.log(f), 1)
    D.append(2 - alpha)
```

[70]:
```
array = escape_basins.reshape(len(ks), grid_size ** 2)
prob_0 = np.sum(array == -1, axis=1) / (grid_size ** 2)
prob_1 = np.sum(array == 0, axis=1) / (grid_size ** 2)
prob_2 = np.sum(array == 1, axis=1) / (grid_size ** 2)
```

[71]:
```
ps = PlotStyler(ticks_on_all_sides=False)
ps.apply_style()

# Create figure
fig = plt.figure(figsize=(12, 4))

gs = gridspec.GridSpec(2, 2)

ax = []
ax.append(fig.add_subplot(gs[:, 0]))
ax.append(fig.add_subplot(gs[0, 1]))
ax.append(fig.add_subplot(gs[1, 1]))
ps.set_tick_padding(ax[0], pad_x=8)
ps.set_tick_padding(ax[2], pad_x=8)

width = ks[1] - ks[0]
ax[0].bar(ks, prob_0, label='State 0', linewidth=1., edgecolor='black',␣
 ↪width=width, align='edge', color="green")
ax[0].bar(ks, prob_1, bottom=prob_0, label='State 1', linewidth=1.,␣
 ↪edgecolor='black', width=width, align='edge', color="gold")
ax[0].bar(ks, prob_2, bottom=prob_0 + prob_1, label='State 2', linewidth=1.,␣
 ↪edgecolor='black', width=width, align='edge', color="blue")
ax[0].set_xlim(0, 1)
ax[0].set_xlabel("$k$")
```

```
ax[0].set_ylim(0, 1)
ax[0].set_ylabel("Basin stability")

ax[1].plot(ks, Sb, "o-", color="blueviolet", label="$S_b$")
ax[1].plot(ks, Sbb, "o-", color="maroon", label="$S_{bb}$")
ax[1].set_ylabel("Basin entropy")
ax[1].legend(loc="lower right", frameon=False)
ax[1].set_xticklabels([])
ax[1].set_yticks([0, 0.5, 1, 1.5])

ax[2].plot(ks, D, "o-", color="blueviolet")
ax[2].set_ylabel("$d$")
ax[2].set_xlabel("$k$")
ax[2].set_yticks([1.4, 1.6, 1.8, 2])

[ax[i].set_xlim(min(ks), max(ks)) for i in range(len(ax))]

plt.subplots_adjust(left=0.056, bottom=0.145, right=0.987, top=0.978, wspace=0.
 ↪14, hspace=0.1)

plt.savefig("fig14.png", dpi=400)
```

```
<Figure size 640x480 with 0 Axes>
```