



NLD

NONLINEAR DYNAMICS LABORATORY

When amplitude changes the story

In nonlinear structures, amplitude changes frequency, mode shape and stability.



— LAB GUIDE NLD-G-02

Figure preparation guide

One figure style in Python, MATLAB and Mathematica

Nonlinear Dynamics Laboratory

October 2026

Summary

A figure carries one result. A reader should grasp it at print size without the text. This guide sets one figure style for the lab and implements it in three tools. The same data give the same figure in Python, MATLAB and Mathematica.

Each tool has a small style file:

- Python: `nld_figstyle.py` (matplotlib);
- MATLAB: `nld_figstyle.m`, `nld_figure.m`, `nld_save.m`;
- Mathematica: `NLDFigureStyle.wl`.

The files, the example data and the example scripts are in the lab drive folder *NLD Templates / NLD brand 2026 / Figure guide*. Section 1 lists the rules. Section 2 shows how to load the style. Section 3 works through four examples in all three tools. The appendix lists the style files in full.

Contents

1 Rules	3
1.1 Size first	3
1.2 Type	4
1.3 Lines, markers and axes	4
1.4 Colour	4
1.5 Measured, computed and analytical results	5
1.6 Export	5
2 Setting up each tool	5
2.1 Python	5
2.2 MATLAB	6
2.3 Mathematica	6
3 Worked examples	7
3.1 E1. Time series	8
3.2 E2. Frequency response with backbone curve	9

3.3 E3. Poincaré section	10
3.4 E4. Multi-panel figure	12
4 Checklist before submission	13
A Style files	14
A.1 Python: <code>nld_figstyle.py</code>	14
A.2 MATLAB: <code>nld_figstyle.m</code>	15
A.3 MATLAB: <code>nld_figure.m</code>	16
A.4 MATLAB: <code>nld_save.m</code>	16
A.5 Mathematica: <code>NLDFigureStyle.wl</code>	17
B Example scripts	18
B.1 Python: <code>examples.py</code>	18
B.2 MATLAB: <code>examples_nld.m</code>	19
B.3 Mathematica: <code>examples_nld.wl</code>	21

1 Rules

1.1 Size first

Make the figure at its final printed size. Never draw large and let the journal shrink it. Shrinking thins the lines and makes the 8 pt tick labels unreadable. The style files take a width keyword and a height in millimetres.

Table 1 Figure widths. The values follow Elsevier artwork sizes; check the target journal.

Keyword	Width	Use
single	90 mm	one column; most figures
onehalf	140 mm	one and a half columns
double	190 mm	full page width; multi-panel figures

Other publishers differ. Springer journals, for example, use 84 mm and 174 mm columns. Pass a number in millimetres when a journal needs another width. Choose the height from the content: 50–60 mm for a single-column line plot, equal axes for phase portraits and sections.

1.2 Type

- Computer Modern in every figure, so labels match the equations in the paper.
- Axis labels 9 pt; tick labels and legends 8 pt. Nothing below 7 pt after placement.
- Symbols in math italic, units upright in parentheses: x (V), t (s), Ω/ω_0 . Use the same symbol as the text and the equations.
- Non-dimensional quantities carry no unit. Do not write “(–)” or “(a.u.)”.
- No title above the axes. The caption states the message.
- Sentence case in labels and legends.

1.3 Lines, markers and axes

- Data lines 1.25 pt. Axes, ticks and reference curves 0.75–1.0 pt.
- Ticks point inward on all four sides. No grid, unless values must be read off the plot.
- Legends without a frame, placed inside the axes where they cover no data. Label curves directly when there are only two or three.
- Dense point sets (Poincaré sections, delay plots) use small filled dots without edges.
- Panel labels (a), (b), ... sit at the upper left, outside the axes, in 9 pt.

1.4 Colour

The NLD figure palette has five series colours, one reference colour and one axis colour. Series take the colours in order. The order is fixed so the same quantity keeps the same colour across the figures of one paper.

Table 2 NLD figure palette (design-system tokens).

	Token	Hex	Use
	fig-1	#1F5A85	first series (blue)
	fig-2	#B4622A	second series (copper)
	fig-3	#588A75	third series (sage)
	fig-4	#CA863E	fourth series (ochre); lines ≥ 1.5 pt
	fig-5	#774860	fifth series (plum)
	fig-ref	#8A8578	reference and linear-model curves; always dashed
	fig-axis	#132433	axes, ticks, labels, annotations

- Never let colour carry meaning alone. Pair it with line style, marker or a direct label. Stable and unstable branches differ in line style, not only in colour.
- More than five series in one axes is too many. Split the figure or use small multiples.
- Ochre is light. Use it for thick lines or filled areas, not thin lines or text.

- Check each figure in greyscale and with a colour-vision simulator (for example [Color Oracle](#)).
- For continuous fields use a perceptually uniform map (viridis, cividis). Do not use jet or rainbow maps.

1.5 Measured, computed and analytical results

Readers must tell measurement from model at a glance.

- Measured data: markers, or thin lines for densely sampled signals.
- Numerical results: solid lines.
- Analytical approximations and linear models: fig-ref, dashed.
- Unstable solution branches: dotted, same colour as the stable branch.
- Show uncertainty as error bars or a light band. The caption states what it is (standard deviation, confidence interval) and the number of repeats.

State in the caption where the data came from: experiment, simulation or theory.

1.6 Export

- Papers: vector PDF with embedded fonts. Every style file does this.
- Slides and the web: PNG at 300 dpi. Use 600 dpi for line art printed as a raster.
- Never JPEG for plots; compression smears lines and text.
- A section with 10^5 points makes a slow PDF. Rasterize only the dots at 600 dpi (`rasterized=True` in matplotlib) and keep axes and text as vectors.
- Before submission, print the page at 100 % and read every label.

2 Setting up each tool

2.1 Python

Requires matplotlib and a LaTeX installation (MacTeX or TeX Live) for true LaTeX text. Without LaTeX, `setup()` falls back to matplotlib's Computer Modern mathtext.

```
import nld_figstyle as nld
nld.setup()                                # LaTeX text, NLD palette, sizes
fig, ax = nld.figure("single", 55)         # 90 mm wide, 55 mm tall
ax.plot(t, x)                              # series take fig-1 ... fig-5 in order
ax.set_xlabel(r"$t$ (s)"); ax.set_ylabel(r"$x$ (V)")
nld.save(fig, "timeseries")                # timeseries.pdf + timeseries.png
```

Keep `nld_figstyle.py` next to the script or on PYTHONPATH.

2.2 MATLAB

Put the three files on the MATLAB path. `nld_figstyle` sets root defaults, so it affects every figure made afterwards in the session.

```
S = nld_figstyle();           % LaTeX text, palette, sizes; S holds colours
fig = nld_figure('single', 55); % 90 x 55 mm, axes ready
plot(t, x);
xlabel('$t$ (s)'); ylabel('$x$ (V)');
nld_save(fig, 'timeseries');  % vector PDF + 300 dpi PNG
```

`nld_save` uses `exportgraphics` with 'Padding', 'figure' (R2025a and later) to keep the exact page size. Older releases crop to the content. It also typesets numeric tick labels in math mode, which gives true minus signs, and stops MATLAB from shrinking the 8 pt tick labels in small axes. MATLAB still shrinks crowded tick labels on its own, so keep ticks sparse (`xticks(0:1:3)`). An equal-aspect panel cannot fill a `tiledlayout` tile; place such panels with axes and explicit positions, as in E4.

2.3 Mathematica

```
Get["NLDFigureStyle.wl"]
p = ListLinePlot[data, nldOptions["single", 55],
  FrameLabel -> {nldLabel["t", "s"], nldLabel["x", "V"]});
nldSave[p, "timeseries"] (* vector PDF + 300 dpi PNG *)
```

Text uses CMU Serif (Computer Modern) when it is installed, otherwise Times. Install CMU Serif from [Font Squirrel](#). For labels typeset by LaTeX itself, use the [MaTeX](#) package. In a script kernel (wolframscript) `$FontFamilies` is empty, so the package assumes CMU Serif is installed. `nldSave` exports with `Magnification -> 1`; without it, the PDF page shrinks to 72 % of `ImageSize`. Options listed before `nldOptions[...]` override it; options listed after it are ignored. Put `AspectRatio`, `ImageSize` and `PlotRange` first when they must change.

3 Worked examples

The examples use lab data. Each one shows the Python version at true size, then the MATLAB and Mathematica versions reduced to fit the page. The scripts are `examples.py`, `examples_nld.m` and `examples_nld.wl`.

Data.

- `timeseries.csv`, `trajectory.csv`, `section.csv`: measured response of the NLD two-well magneto-elastic oscillator (2002), sampled at 160 Hz, 16 samples per forcing period. The delay plot uses $\tau = 4$ samples; the section takes every 16th sample.
- `frf.csv`, `backbone.csv`: harmonic-balance solution of the Duffing oscillator $\ddot{x} + 2\zeta\dot{x} + x + \gamma x^3 = F \cos \Omega t$ with $\zeta = 0.05$, $\gamma = 1$, for three forcing levels. Column branch: 0 upper or single, 1 unstable, 2 lower. The backbone is $\Omega = \sqrt{1 + \frac{3}{4}\gamma A^2}$.

3.1 E1. Time series

Single column, 90×50 mm. One series in fig-1. The time axis starts at the first sample and ends at the last; no padding.

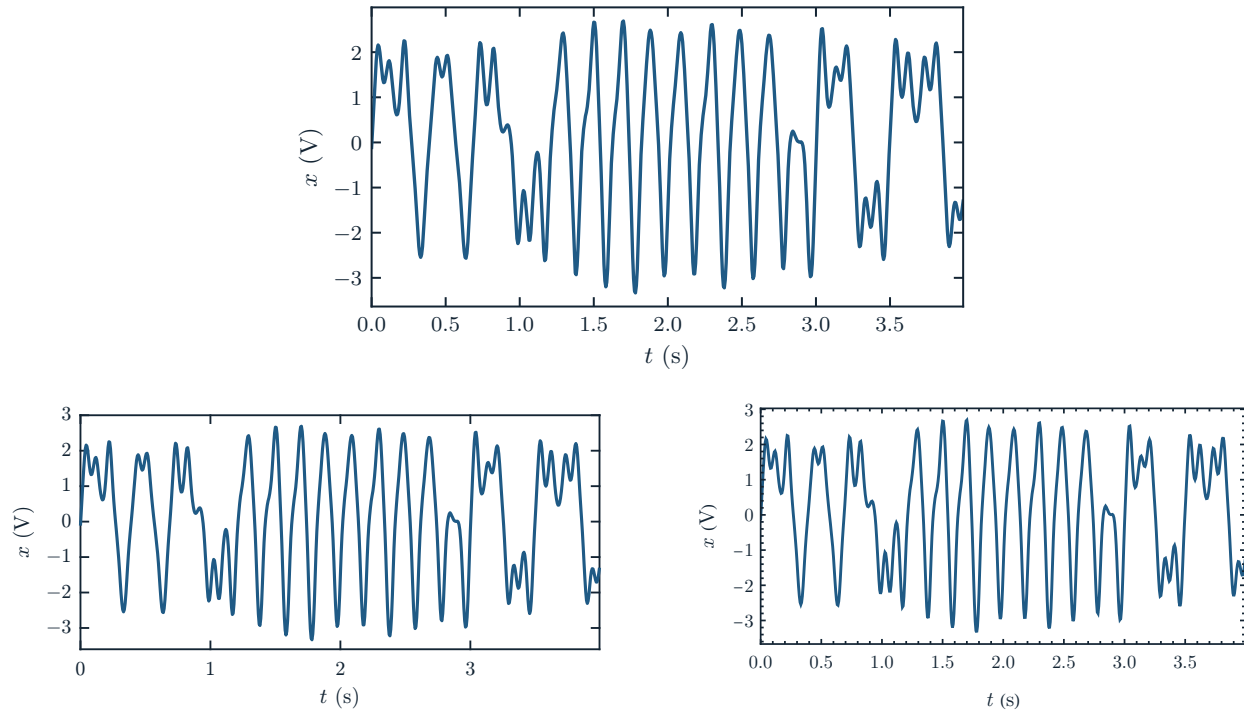


Figure 1 Measured two-well oscillator response. Top: Python, true size. Bottom: MATLAB (left) and Mathematica (right), reduced to 83 %.

Python

```
fig, ax = nld.figure("single", 50)
ax.plot(ts[:, 0], ts[:, 1])
ax.set_xlabel(r"$t$ (s)"); ax.set_ylabel(r"$x$ (V)")
ax.set_xlim(0, ts[-1, 0])
```

MATLAB

```
fig = nld_figure('single', 50);
plot(ts(:,1), ts(:,2)); xlim([0 ts(end,1)]); xticks(0:1:3);
xlabel('$t$ (s)'); ylabel('$x$ (V)');
```

Mathematica

```
e1 = ListLinePlot[ts, nldOptions["single", 50], PlotRange -> {{0, ts[[-1, 1]]}, All},
  FrameLabel -> {nldLabel["t", "s"], nldLabel["x", "V"]}];
```

3.2 E2. Frequency response with backbone curve

Single column, 90×60 mm. Each forcing level takes one palette colour. Stable branches are solid; the unstable branch is dotted in the same colour. The backbone curve is the reference: fig-ref, dashed. The legend names the forcing levels, so the reader does not decode colour alone.

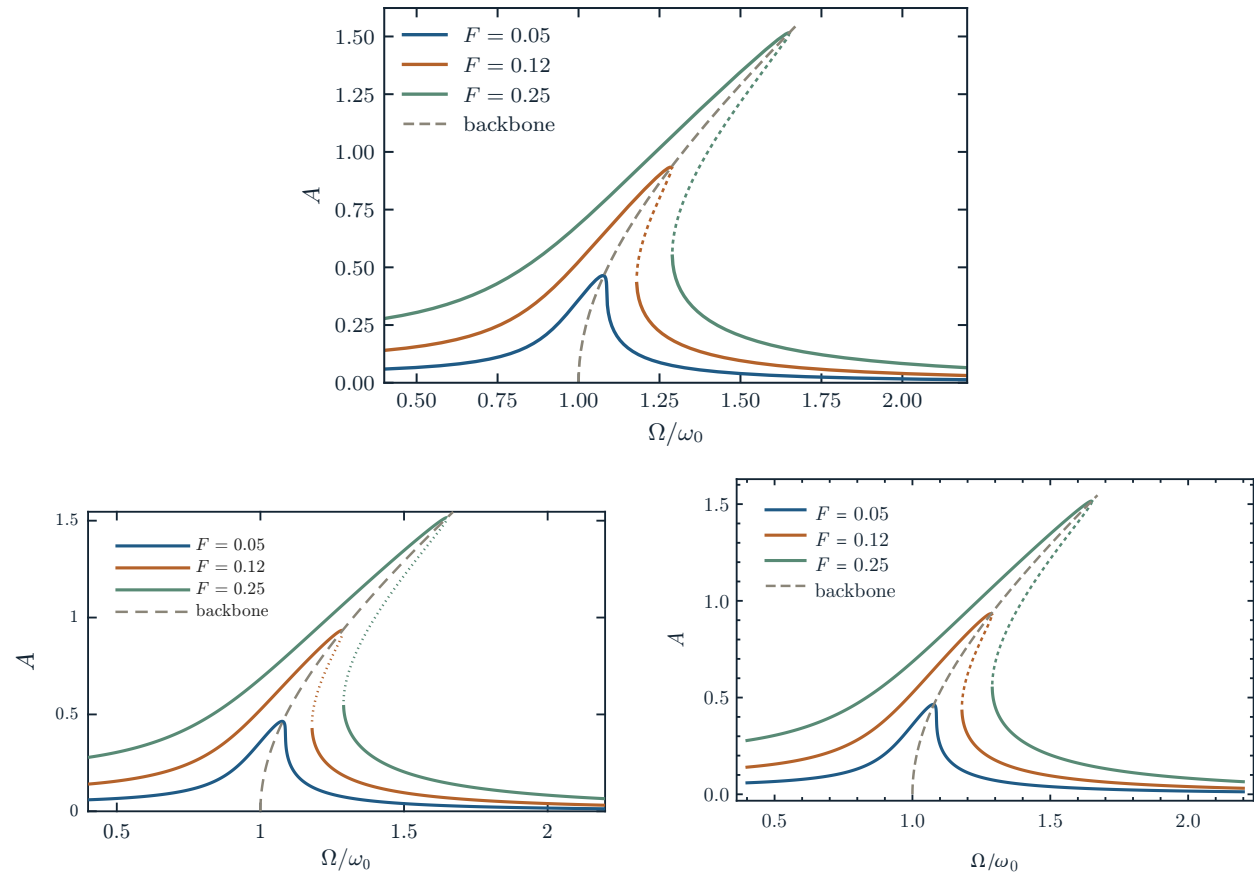


Figure 2 Duffing frequency response at three forcing levels with the backbone curve (harmonic balance). Top: Python, true size. Bottom: MATLAB (left) and Mathematica (right), reduced to 83 %.

Python

```
for i, F in enumerate(np.unique(frf[:, 0])):
    for br in (0, 2, 1):
        s = frf[(frf[:, 0] == F) & (frf[:, 1] == br)]
        ax.plot(s[:, 2], s[:, 3], color=nld.COLORS[i],
                ls="-" if br != 1 else (0, (1.5, 1.5)),
                label=rf"$F={F:g}$" if br == 0 else None)
nld.ref_line(ax, bb[:, 0], bb[:, 1], label="backbone")
```

MATLAB

```
if br == 1, plot(s(:,3), s(:,4), ':', 'Color',S.colors(i,:), 'LineWidth',1.0);
else,      p = plot(s(:,3), s(:,4), '-', 'Color',S.colors(i,:)); end
hb = plot(bb(:,1), bb(:,2), '--', 'Color',S.ref, 'LineWidth',S.thin+0.25);
```

```
| legend([h; hb], [compose('$F=%g$',Fs); {'backbone'}], 'Location','northwest');
```

Mathematica

```
| ListLinePlot[branch[F, 1], PlotStyle -> Directive[c, AbsoluteThickness[1.0],  
| AbsoluteDashing[{1.5, 1.5}]]]  
| ListLinePlot[bb, PlotStyle -> Directive[nldRef, AbsoluteThickness[1.0],  
| AbsoluteDashing[{4, 2}]]]
```

3.3 E3. Poincaré section

Single column, 90×80 mm, equal axis scales. Six thousand strobed points as small filled dots without edges. Equal scales keep the geometry of the attractor. All three tools use the same limits, $x_n \in [-3, 3]$ and $x_{n+\tau} \in [-3, 3.5]$. Set the limits explicitly: MATLAB's `axis equal tight` trims the limits to the data and clips edge points. Mathematica fixes the height and lets the width follow from equal scales, so its figure is narrower than 90 mm.

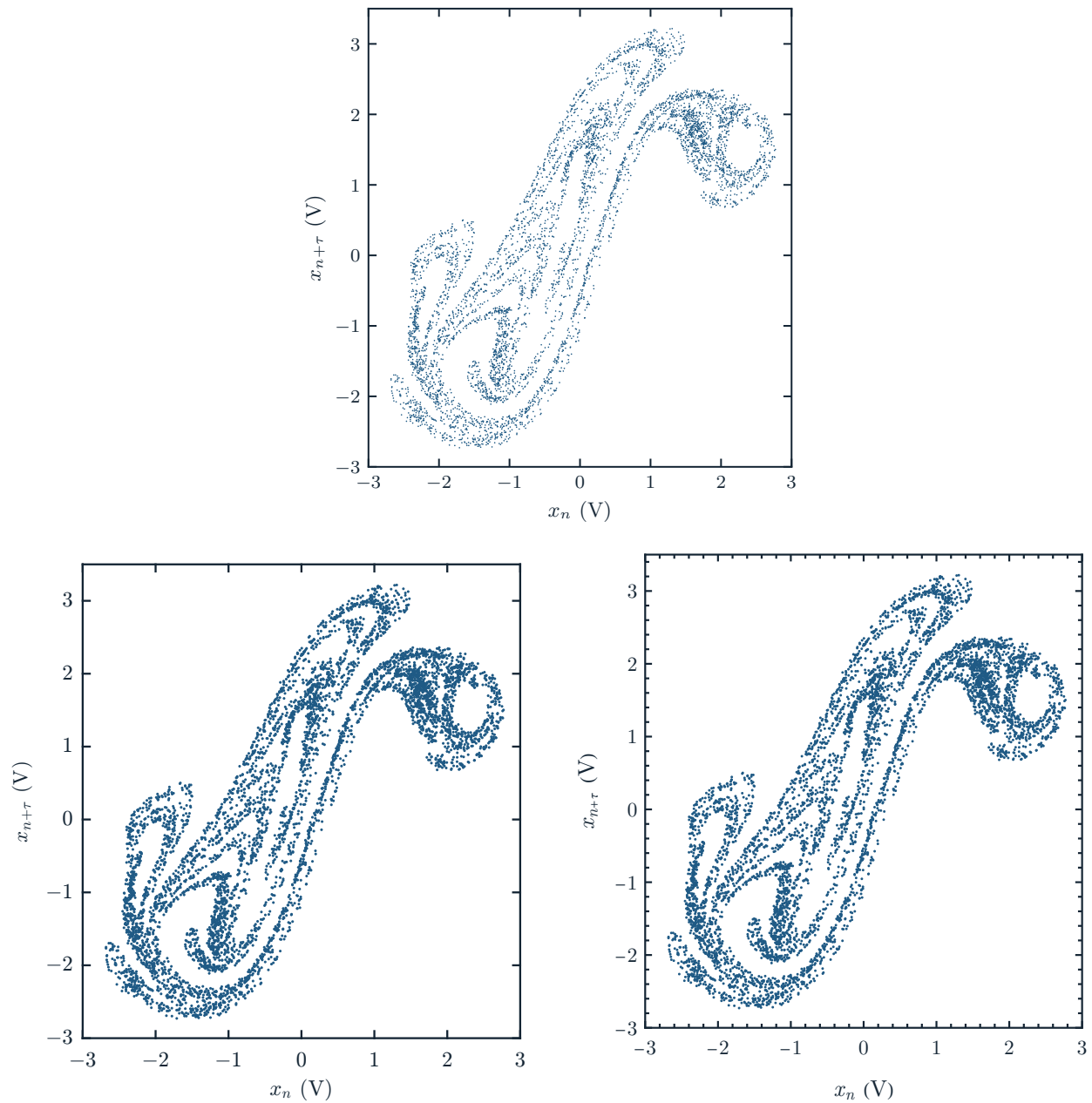


Figure 3 Poincaré section of the measured two-well response in delay coordinates. Top: Python, true size. Bottom: MATLAB (left) and Mathematica (right), reduced to 73 %.

Python

```
ax.plot(sec[:, 0], sec[:, 1], ".", ms=1.2, mew=0, color=nld.COLORS[0])
ax.set_xlim(-3, 3); ax.set_ylim(-3, 3.5); ax.set_aspect("equal")
```

MATLAB

```
scatter(sec(:,1), sec(:,2), 1, S.colors(1,:), 'filled');
xlim([-3 3]); ylim([-3 3.5]); daspect([1 1 1]);
```

Mathematica

```
ListPlot[sec, AspectRatio -> Automatic, ImageSize -> {Automatic, nldWidth[80]},  
PlotRange -> {{-3, 3}, {-3, 3.5}}, nldOptions["single", 80],  
PlotStyle -> Directive[nldColors[[1]], AbsolutePointSize[1]]]
```

3.4 E4. Multi-panel figure

Double column, 190×65 mm. Panel (a) gets the larger share of the width for the time series; panel (b) keeps equal axes. The orbit is drawn thin and light so the copper section points stand out. Panel labels sit outside the axes at the upper left.

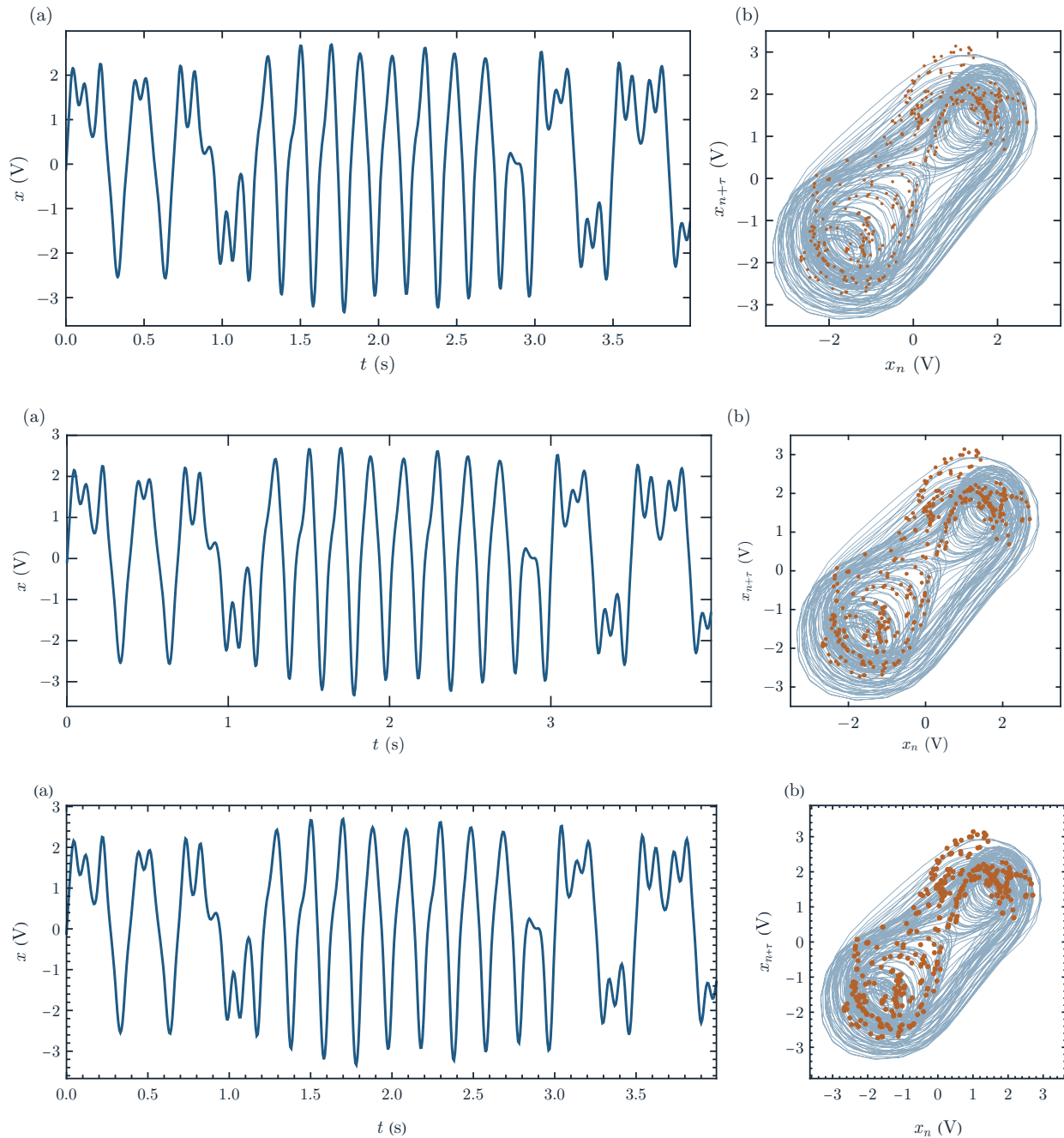


Figure 4 (a) Measured response; (b) delay reconstruction with Poincaré points. Python, MATLAB and Mathematica from top to bottom, each reduced to 87 %.

Python

```
fig, (a, b) = nld.figure("double", 65, 1, 2, width_ratios=[1.6, 1])
b.plot(trj[:, 0], trj[:, 1], lw=0.4, color=nld.COLORS[0], alpha=0.5)
b.plot(sec[:400, 0], sec[:400, 1], ".", ms=3, mew=0, color=nld.COLORS[1])
nld.panel_label(a, "(a)"); nld.panel_label(b, "(b)")
```

MATLAB

```
s = Hf - m.bottom - m.top; % side of the square panel (pt)
a = axes(fig, 'Units', 'points', 'Position', [m.left m.bottom W-m.right-s-m.gap-m.left s
]);
b = axes(fig, 'Units', 'points', 'Position', [W-m.right-s m.bottom s s]);
title(a, '(a)', 'FontWeight', 'normal', 'HorizontalAlignment', 'left', ...
      'Units', 'normalized', 'Position', [-0.07 1.02], 'VerticalAlignment', 'bottom');
```

Mathematica

```
a = nldPanel[ListLinePlot[ts, nldOptions[129, 65], ...], "(a)"];
b = nldPanel[Show[... , AspectRatio -> Automatic, ImageSize -> {Automatic, nldWidth
[65]},
PlotRange -> {{-3.5, 3.5}, {-3.5, 3.5}}, nldOptions[72, 65], ...], "(b)"];
e4 = Grid[{{a, b}}, Spacings -> 0];
```

4 Checklist before submission

- Made at final width; height suits the content.
- Labels 9 pt, ticks 8 pt, Computer Modern; symbols match the text.
- Every axis has a quantity and a unit, or is clearly non-dimensional.
- Palette in order; fig-ref dashed for reference curves; no colour-only encoding.
- Measured, numerical and analytical results look different and are named in the caption.
- Uncertainty shown and defined, with the number of repeats.
- Panel labels (a), (b), ... present and cited in the caption.
- Vector PDF with embedded fonts; dense dots rasterized at 600 dpi if needed.
- Readable in greyscale and in a colour-vision simulation.
- Printed at 100 % and read.

A Style files

A.1 Python: nld_figstyle.py

```
"""NLD figure style for matplotlib.

import nld_figstyle as nld
nld.setup()                    # LaTeX text, NLD palette, sizes
fig, ax = nld.figure("single", 55) # 90 mm wide, 55 mm tall
ax.plot(t, x)                  # series take fig-1 ... fig-5 in order
nld.save(fig, "timeseries")    # vector PDF (and a 300 dpi PNG)
```

Requires a LaTeX installation (MacTeX, TeX Live). Without one, setup()
falls back to matplotlib's Computer Modern mathtext.

```
"""
import os
import shutil
import matplotlib as mpl
import matplotlib.pyplot as plt

# NLD figure palette (design system tokens fig-1 ... fig-5, fig-ref, fig-axis)
COLORS = ["#1F5A85", "#B4622A", "#588A75", "#CA863E", "#774860"]
REF = "#8A8578"      # reference / linear-model curves, always dashed
AXIS = "#132433"     # axes, ticks, labels, annotation text

MM = 1 / 25.4        # inches per millimetre
WIDTH = {"single": 90, "onehalf": 140, "double": 190} # mm; common journal column widths

LINE = 1.25          # pt, data lines (use 1.5 for fig-4 ochre)
THIN = 0.75          # pt, reference lines, axes

def setup(usetex=True):
    """Apply the NLD style to every figure made afterwards."""
    if usetex and shutil.which("latex") is None and os.path.isdir("/Library/TeX/
texbin"):
        os.environ["PATH"] += os.pathsep + "/Library/TeX/texbin"
    usetex = usetex and shutil.which("latex") is not None
    mpl.rcParams.update({
        "text.usetex": usetex,
        "font.family": "serif",
        "font.serif": ["Computer Modern Roman"],
        "mathtext.fontset": "cm",
        "text.latex.preamble": r"\usepackage{amsmath}",
        "font.size": 9, "axes.labelsize": 9, "xtick.labelsize": 8,
        "ytick.labelsize": 8, "legend.fontsize": 8,
        "axes.prop_cycle": mpl.cycler(color=COLORS),
        "lines.linewidth": LINE,
        "axes.linewidth": THIN, "axes.edgecolor": AXIS, "axes.labelcolor": AXIS,
        "xtick.color": AXIS, "ytick.color": AXIS, "text.color": AXIS,
        "xtick.major.width": THIN, "ytick.major.width": THIN,
        "xtick.direction": "in", "ytick.direction": "in",
        "xtick.top": True, "ytick.right": True,
```

```

        "axes.spines.top": True, "axes.spines.right": True,
        "axes.grid": False,
        "legend.frameon": False,
        "figure.facecolor": "white", "axes.facecolor": "white",
        "savefig.bbox": None, # keep the exact journal width
        "pdf.fonttype": 42,
    })
    return usetex

def figure(width="single", height_mm=55, nrows=1, ncols=1, **kw):
    """Figure at a journal width ('single', 'onehalf', 'double' or mm)."""
    w = WIDTH.get(width, width) if isinstance(width, str) else width
    return plt.subplots(nrows, ncols, figsize=(w * MM, height_mm * MM),
                        constrained_layout=True, **kw)

def ref_line(ax, x, y, **kw):
    """Reference or linear-model curve: fig-ref, dashed, thin."""
    return ax.plot(x, y, color=REF, ls=(0, (4, 2)), lw=THIN + 0.25, **kw)

def panel_label(ax, label):
    """Panel label '(a)' in the upper-left corner, outside the data."""
    ax.text(-0.02, 1.02, label, transform=ax.transAxes, ha="right", va="bottom")

def save(fig, name, outdir=".", png=True):
    """Vector PDF for papers; 300 dpi PNG for slides and the web."""
    fig.savefig(os.path.join(outdir, name + ".pdf"))
    if png:
        fig.savefig(os.path.join(outdir, name + ".png"), dpi=300)

```

A.2 MATLAB: nld_figstyle.m

```

function S = nld_figstyle()
%NLD_FIGSTYLE Apply the NLD figure style to every figure made afterwards.
% S = nld_figstyle() sets LaTeX text (Computer Modern), the NLD figure
% palette, font sizes and line widths as root defaults, and returns a
% struct with the colours and sizes for use in plotting code.
%
% S.colors 5x3 RGB, NLD palette fig-1 ... fig-5 (series order)
% S.ref    fig-ref, reference / linear-model curves (always dashed)
% S.axis   fig-axis, axes, ticks, labels and annotation text
% S.width  journal widths in mm: single 90, onehalf 140, double 190
%
% See also NLD_FIGURE, NLD_SAVE.

S.colors = [ 31  90 133;    % fig-1 blue
            180 98  42;    % fig-2 copper
            88 138 117;    % fig-3 sage
            202 134 62;    % fig-4 ochre (use >= 1.5 pt)
            119 72  96]/255;% fig-5 plum
S.ref    = [138 133 120]/255;

```

```

S.axis = [ 19 36 51]/255;
S.width = struct('single',90,'onehalf',140,'double',190);
S.line = 1.25; % pt, data lines
S.thin = 0.75; % pt, reference lines and axes

set(groot,'defaultTextInterpreter','latex', ...
    'defaultAxesTickLabelInterpreter','latex', ...
    'defaultLegendInterpreter','latex', ...
    'defaultColorbarTickLabelInterpreter','latex', ...
    'defaultAxesFontSize',8, 'defaultTextFontSize',9, ...
    'defaultAxesLabelFontSizeMultiplier',9/8, ...
    'defaultLegendFontSize',8, 'defaultLegendBox','off', ...
    'defaultAxesColorOrder',S.colors, ...
    'defaultAxesLineWidth',S.thin, ...
    'defaultAxesXColor',S.axis,'defaultAxesYColor',S.axis,'defaultAxesZColor',S
    .axis, ...
    'defaultTextColor',S.axis, ...
    'defaultAxesTickDir','in','defaultAxesBox','on', ...
    'defaultAxesTickLength',[0.015 0.025], ...
    'defaultLineLineWidth',S.line, ...
    'defaultFigureColor','w','defaultAxesColor','w');

end

```

A.3 MATLAB: nld_figure.m

```

function [fig, t] = nld_figure(width, height_mm, m, n)
%NLD_FIGURE New figure at a journal width, laid out edge to edge.
% fig = nld_figure('single', 55)          90 x 55 mm, one axes ready to plot
% [fig, t] = nld_figure('double', 65, 1, 2) 1x2 tiled layout; call nexttile(t)
% width: 'single' (90 mm), 'onehalf' (140 mm), 'double' (190 mm) or mm.
if ischar(width) || isstring(width)
    w = struct('single',90,'onehalf',140,'double',190).(char(width));
else
    w = width;
end
if nargin < 3, m = 1; n = 1; end
fig = figure('Units','centimeters','Position',[2 2 w/10 height_mm/10], 'Color','w');
t = tiledlayout(fig, m, n, 'TileSpacing','compact', 'Padding','compact'); % 'tight'
    clips the box edge
if m*n == 1, nexttile(t); end
end

```

A.4 MATLAB: nld_save.m

```

function nld_save(fig, name, outdir)
%NLD_SAVE Vector PDF for papers and a 300 dpi PNG for slides and the web.
% nld_save(gcf, 'timeseries') writes timeseries.pdf and .png
% The exported page keeps the figure size set by NLD_FIGURE.
if nargin < 3, outdir = '.'; end
% Typeset numeric tick labels in math mode: true minus signs, same font as the text
for ax = findall(fig, 'Type', 'axes')
    % MATLAB shrinks fonts in small axes; keep the style's 8 pt ticks and 9 pt labels
    if strcmp(ax.FontSizeMode, 'auto'), ax.FontSize = get(groot, 'defaultAxesFontSize'); end
end

```

```

    for r = [ax.XAxis, ax.YAxis]
        if isa(r, 'matlab.graphics.axis.decorator.NumericRuler')
            r.TickLabelFormat = '$%g$';
            r.FontSize = ax.FontSize; % rulers can shrink their labels on their own
        end
    end
end
end
opts = {'BackgroundColor','white'};
try % R2025a+: keep the full figure size instead of cropping
    exportgraphics(fig, fullfile(outdir,[name '.pdf']), 'ContentType','vector', '
    Padding','figure', opts{:});
    exportgraphics(fig, fullfile(outdir,[name '.png']), 'Resolution',300, 'Padding','
    figure', opts{:});
catch
    exportgraphics(fig, fullfile(outdir,[name '.pdf']), 'ContentType','vector', opts
    {:});
    exportgraphics(fig, fullfile(outdir,[name '.png']), 'Resolution',300, opts{:});
end
end

```

A.5 Mathematica: NLDFigureStyle.wl

(* NLDFigureStyle.wl -- NLD figure style for Mathematica.

```

Get["NLDFigureStyle.wl"]
p = ListLinePlot[data, nldOptions["single", 50],
    FrameLabel -> {nldLabel["t", "s"], nldLabel["x", "V"]}};
nldSave[p, "timeseries"] (* vector PDF + 300 dpi PNG *)

```

Text uses CMU Serif (Computer Modern) when it is installed, otherwise Times.
 Install CMU Serif from <https://www.fontsquirrel.com/fonts/computer-modern>
 or use the MaTeX package for true LaTeX labels. *)

```

BeginPackage["NLDFigureStyle`"];

nldColors::usage = "NLD figure palette fig-1 ... fig-5, in series order.";
nldRef::usage = "fig-ref colour for reference and linear-model curves (always dashed)
.";
nldAxis::usage = "fig-axis colour for frames, ticks and labels.";
nldFont::usage = "Font family used for figure text.";
nldWidth::usage = "nldWidth[\"single\"|\"onehalf\"|\"double\"] gives the width in
    printer points.";
nldOptions::usage = "nldOptions[width, heightMM] gives plot options for an NLD figure
.";
nldLabel::usage = "nldLabel[\"x\", \"V\"] gives an italic symbol with its unit: x (V)
.";
nldPanel::usage = "nldPanel[g, \"(a)\"] adds a panel label to graphic g.";
nldSave::usage = "nldSave[g, name] exports name.pdf (vector) and name.png (300 dpi).";
;

Begin["`Private`"];

nldColors = RGBColor /@ {"#1F5A85", "#B4622A", "#588A75", "#CA863E", "#774860"};
nldRef = RGBColor["#8A8578"];

```

```

nldAxis = RGBColor["#132433"];
(* $FontFamilies is empty in a script kernel; then assume CMU Serif is installed *)
nldFont = If[$FontFamilies === {} || MemberQ[$FontFamilies, "CMU Serif"], "CMU Serif"
, "Times"];

mm = 72/25.4; (* printer points per millimetre *)
nldWidth[w_String] := mm Lookup[<|"single" -> 90, "onehalf" -> 140, "double" ->
190|>, w];
nldWidth[w_?NumericQ] := mm w;

nldOptions[w_, hmm_] := {
  Frame -> True, Axes -> False,
  FrameStyle -> Directive[nldAxis, AbsoluteThickness[0.75]],
  FrameTicksStyle -> Directive[nldAxis, FontFamily -> nldFont, 8],
  LabelStyle -> Directive[nldAxis, FontFamily -> nldFont, 9],
  PlotStyle -> (Directive[#, AbsoluteThickness[1.25]] & /@ nldColors),
  ImageSize -> {nldWidth[w], mm hmm}, AspectRatio -> Full,
  ImagePadding -> {{38, 8}, {34, 6}}, Background -> White};

nldLabel[sym_, unit_String] := Row[{Style[sym, Italic], " (" , unit, ")" }];
nldLabel[sym_] := Style[sym, Italic];

nldPanel[g_, lab_String] := Show[g, Epilog -> Inset[Style[lab, FontFamily -> nldFont,
9, nldAxis],
  Scaled[{-0.02, 1.02}], {Right, Bottom}], PlotRangeClipping -> False,
  ImagePadding -> {{38, 8}, {34, 16}}];

(* Magnification -> 1 keeps the exported page at ImageSize; without it the PDF
shrinks to 72 % *)
nldSave[g_, name_String, dir_String : "."] := With[{gm = Style[g, Magnification ->
1]},
  Export[FileNameJoin[{dir, name <> ".pdf"}], gm, "AllowRasterization" -> False];
  Export[FileNameJoin[{dir, name <> ".png"}], gm, ImageResolution -> 300]];

End[];
EndPackage[];

```

B Example scripts

B.1 Python: examples.py

```

"""The four guide examples in matplotlib. Run from this folder: python examples.py"""
import numpy as np
import nld_figstyle as nld

D, OUT = "../data/", "../out/"
nld.setup()

ts = np.loadtxt(D + "timeseries.csv", delimiter=",", skiprows=1)
frf = np.loadtxt(D + "frf.csv", delimiter=",", skiprows=1)
bb = np.loadtxt(D + "backbone.csv", delimiter=",", skiprows=1)
sec = np.loadtxt(D + "section.csv", delimiter=",", skiprows=1)
trj = np.loadtxt(D + "trajectory.csv", delimiter=",", skiprows=1)

```

```

# E1 time series
fig, ax = nld.figure("single", 50)
ax.plot(ts[:, 0], ts[:, 1])
ax.set_xlabel(r"$t$ (s)"); ax.set_ylabel(r"$x$ (V)")
ax.set_xlim(0, ts[-1, 0])
nld.save(fig, "python_E1", OUT)

# E2 frequency response with backbone curve
fig, ax = nld.figure("single", 60)
for i, F in enumerate(np.unique(frf[:, 0])):
    c = nld.COLORS[i]
    for br in (0, 2, 1):
        s = frf[(frf[:, 0] == F) & (frf[:, 1] == br)]
        if len(s):
            ax.plot(s[:, 2], s[:, 3], color=c, ls="-" if br != 1 else (0, (1.5, 1.5)))
            lw=nld.LINE if br != 1 else 1.0,
            label=rf"$F={F:g}$" if br == 0 else None)
nld.ref_line(ax, bb[:, 0], bb[:, 1], label="backbone")
ax.set_xlim(0.4, 2.2); ax.set_ylim(0, None)
ax.set_xlabel(r"$\Omega/\omega_0$"); ax.set_ylabel(r"$A$")
ax.legend(loc="upper left")
nld.save(fig, "python_E2", OUT)

# E3 Poincare section
fig, ax = nld.figure("single", 80)
ax.plot(sec[:, 0], sec[:, 1], ".", ms=1.2, mew=0, color=nld.COLORS[0])
ax.set_xlim(-3, 3); ax.set_ylim(-3, 3.5); ax.set_aspect("equal")
ax.set_xlabel(r"$x_n$ (V)"); ax.set_ylabel(r"$x_{n+\tau}$ (V)")
nld.save(fig, "python_E3", OUT)

# E4 two panels, double column
fig, (a, b) = nld.figure("double", 65, 1, 2, width_ratios=[1.6, 1])
a.plot(ts[:, 0], ts[:, 1]); a.set_xlim(0, ts[-1, 0])
a.set_xlabel(r"$t$ (s)"); a.set_ylabel(r"$x$ (V)")
b.plot(trj[:, 0], trj[:, 1], lw=0.4, color=nld.COLORS[0], alpha=0.5)
b.plot(sec[:, 400, 0], sec[:, 400, 1], ".", ms=3, mew=0, color=nld.COLORS[1])
b.set_xlim(-3.5, 3.5); b.set_ylim(-3.5, 3.5); b.set_aspect("equal")
b.set_xlabel(r"$x_n$ (V)"); b.set_ylabel(r"$x_{n+\tau}$ (V)")
nld.panel_label(a, "(a)"); nld.panel_label(b, "(b)")
nld.save(fig, "python_E4", OUT)
print("done")

```

B.2 MATLAB: examples_nld.m

```

%EXAMPLES_NLD The four guide examples in MATLAB. Run from this folder.
S = nld_figstyle();
D = fullfile('..', 'data'); OUT = fullfile('..', 'out');
ts = readmatrix(fullfile(D, 'timeseries.csv'));
frf = readmatrix(fullfile(D, 'frf.csv')); % columns F, branch, Omega, A
bb = readmatrix(fullfile(D, 'backbone.csv'));
sec = readmatrix(fullfile(D, 'section.csv'));
trj = readmatrix(fullfile(D, 'trajectory.csv'));

```

```

%% E1 time series
fig = nld_figure('single',50);
plot(ts(:,1), ts(:,2));
xlabel('$t$ (s)'); ylabel('$x$ (V)'); xlim([0 ts(end,1)]); ylim([-3.6 3]);
xticks(0:1:3); % MATLAB shrinks crowded tick labels below 8 pt; keep ticks sparse
nld_save(fig,'matlab_E1',OUT);

%% E2 frequency response with backbone curve
fig = nld_figure('single',60); hold on
Fs = unique(frf(:,1)); h = gobjects(numel(Fs),1);
for i = 1:numel(Fs)
    for br = [0 2 1]
        s = frf(frf(:,1)==Fs(i) & frf(:,2)==br, :);
        if isempty(s), continue, end
        if br == 1 % unstable branch: dotted, thinner
            plot(s(:,3), s(:,4), ':', 'Color',S.colors(i,:), 'LineWidth',1.0);
        else
            p = plot(s(:,3), s(:,4), '-', 'Color',S.colors(i,:));
            if br == 0, h(i) = p; end
        end
    end
end
hb = plot(bb(:,1), bb(:,2), '--', 'Color',S.ref, 'LineWidth',S.thin+0.25);
xlim([0.4 2.2]); ylim([0 inf]);
xlabel('$\Omega/\omega_0$'); ylabel('$A$');
legend([h; hb], [compose('$F=\%g$',Fs); {'backbone'}], 'Location','northwest');
nld_save(fig,'matlab_E2',OUT);

%% E3 Poincare section
fig = nld_figure('single',80);
scatter(sec(:,1), sec(:,2), 1, S.colors(1,:), 'filled');
xlim([-3 3]); ylim([-3 3.5]); daspect([1 1 1]); % equal scales, same limits in all
tools
xlabel('$x_n$ (V)'); ylabel('$x_{n+\tau}$ (V)');
nld_save(fig,'matlab_E3',OUT);

%% E4 two panels, double column
% An equal-aspect panel cannot fill a tiledlayout tile, which leaves white space
% at the right. Place the two axes by hand instead, in points.
W = S.width.double*72/25.4; Hf = 65*72/25.4; % figure size (pt)
fig = figure('Units','points','Position',[50 50 W Hf],'Color','w');
m = struct('left',32,'bottom',28,'top',20,'right',6,'gap',40); % margins (pt)
s = Hf - m.bottom - m.top; % side of the square
panel (b)
a = axes(fig,'Units','points','Position',[m.left m.bottom W-m.right-s-m.gap-m.left s
]);
plot(ts(:,1), ts(:,2)); xlim([0 ts(end,1)]); ylim([-3.6 3]); xticks(0:1:3);
xlabel('$t$ (s)'); ylabel('$x$ (V)');
title(a, '(a)', 'FontWeight','normal', 'FontSize',9, 'HorizontalAlignment','left', 'Units',
'normalized', 'Position',[-0.07 1.02], 'VerticalAlignment','bottom');
b = axes(fig,'Units','points','Position',[W-m.right-s m.bottom s s]); hold on
plot(trj(:,1), trj(:,2), '-', 'LineWidth',0.4, 'Color',[S.colors(1,:) 0.5]);
scatter(sec(1:400,1), sec(1:400,2), 4, S.colors(2,:), 'filled');

```

```

xlim([-3.5 3.5]); ylim([-3.5 3.5]); % equal ranges in a
    square box: equal scales
xlabel('$x_n$ (V)'); ylabel('$x_{n+\tau}$ (V)');
title(b, '(b)', 'FontWeight', 'normal', 'FontSize', 9, 'HorizontalAlignment', 'left', 'Units',
    'normalized', 'Position', [-0.24 1.02], 'VerticalAlignment', 'bottom');
nld_save(fig, 'matlab_E4', OUT);
close all
disp('done')

```

B.3 Mathematica: examples_nld.wl

```

(* The four guide examples in Mathematica. Run from this folder:
    wolframscript -file examples_nld.wl (or evaluate in a notebook) *)
SetDirectory[DirectoryName[$InputFileName]];
Get["NLDFigureStyle.wl"];
d = "../data/"; out = "../out/";
csv[f_] := Rest[Import[d <> f, "CSV"]];
ts = csv["timeseries.csv"]; frf = csv["frf.csv"]; (* F, branch, Omega, A *)
bb = csv["backbone.csv"]; sec = csv["section.csv"]; trj = csv["trajectory.csv"];

(* E1 time series *)
e1 = ListLinePlot[ts, nldOptions["single", 50],
    FrameLabel -> {nldLabel["t", "s"], nldLabel["x", "V"]}, PlotRange -> {{0, ts[[-1,
    1]]}, All}};
nldSave[e1, "mathematica_E1", out];

(* E2 frequency response with backbone curve *)
Fs = Union[frf[All, 1]];
branch[F_, b_] := Select[frf, #[[1]] == F && #[[2]] == b &][[All, {3, 4}]];
curves = Table[With[{c = nldColors[[i]], F = Fs[[i]]},
    {ListLinePlot[DeleteCases[branch[F, 0], branch[F, 2]], {}}], PlotStyle ->
    Directive[c, AbsoluteThickness[1.25]]],
    If[branch[F, 1] == {}, Nothing,
        ListLinePlot[branch[F, 1], PlotStyle -> Directive[c, AbsoluteThickness[1.0],
            AbsoluteDashing[{1.5, 1.5}]]]],
    {i, Length[Fs]};
e2 = Show[Flatten[curves],
    ListLinePlot[bb, PlotStyle -> Directive[nldRef, AbsoluteThickness[1.0],
        AbsoluteDashing[{4, 2}]]],
    nldOptions["single", 60], PlotRange -> {{0.4, 2.2}, {0, All}},
    FrameLabel -> {Row[{Style["\[CapitalOmega]", Italic], "/"}, Subscript[Style["\[
    Omega]", Italic], 0]}], nldLabel["A"]},
    Epilog -> Inset[LineLegend[
        Append[Directive[#, AbsoluteThickness[1.25]] & /@ nldColors[;; Length[Fs]],
            Directive[nldRef, AbsoluteThickness[1.0], AbsoluteDashing[{4, 2}]]],
        Append[Row[{Style["F", Italic], " = ", #}] & /@ Fs, "backbone"],
        LabelStyle -> Directive[nldAxis, FontFamily -> nldFont, 8],
        LegendMarkerSize -> {18, 8}, Spacings -> {0.3, 0.1}, Scaled[{0.03, 0.97}], {
            Left, Top}]];
nldSave[e2, "mathematica_E2", out];

(* E3 Poincare section *)
(* Options given before nldOptions take precedence: equal scales need AspectRatio ->
    Automatic

```

```

    and a height-only ImageSize *)
e3 = ListPlot[sec, AspectRatio -> Automatic, ImageSize -> {Automatic, nldWidth[80]},
  PlotRange -> {{-3, 3}, {-3, 3.5}}, nldOptions["single", 80],
  PlotStyle -> Directive[nldColors[[1]], AbsolutePointSize[1]],
  FrameLabel -> {nldLabel[Subscript["x", "n"], "V"], nldLabel[Subscript["x", "n+\[
    Tau]"], "V"]}}];
nldSave[e3, "mathematica_E3", out];

(* E4 two panels, double column *)
a = nldPanel[ListLinePlot[ts, nldOptions[129, 65], PlotRange -> {{0, ts[[-1, 1]]},
  All},
  FrameLabel -> {nldLabel["t", "s"], nldLabel["x", "V"]}, "(a)"];
b = nldPanel[Show[ListLinePlot[trj, PlotStyle -> Directive[Opacity[0.5], nldColors
  [[1]], AbsoluteThickness[0.4]]],
  ListPlot[sec[;; 400], PlotStyle -> Directive[nldColors[[2]], AbsolutePointSize
  [2.5]]],
  AspectRatio -> Automatic, ImageSize -> {Automatic, nldWidth[65]},
  PlotRange -> {{-3.5, 3.5}, {-3.5, 3.5}}, nldOptions[72, 65],
  FrameLabel -> {nldLabel[Subscript["x", "n"], "V"], nldLabel[Subscript["x", "n+\[
    Tau]"], "V"]}, "(b)"];
e4 = Grid[{{a, b}}, Spacings -> 0];
nldSave[e4, "mathematica_E4", out];
Print["done"];

```