Stochastic DEQs on LUCIDAC

1 Introduction

About a year ago we published an application note describing the implementation of a stochastic differential equation (*SDE*)[1] of the form[2]

$$\dot{y} = \lambda y + \sigma y W \tag{1}$$

on THE ANALOG THING connected to a simple Arduino based hybrid controller.[3] This application note shows how this can be done on a LUCIDAC using an external analog noise source. The mechanization is similar to that described in application note #50 as shown in figure 1. Since coefficients in the LUCIDAC can be positive as well as negative, the inverter on the right side is no longer necessary since the feedback coefficient can be set to $-\lambda$. Also, LUCIDAC integrators do not have input weights of 10 and there are coefficients at every computing element input due to the overall system architecture.

2 Implementation

The overall setup is shown in figure 2. On the bottom is a Wandel&Goltermann RG-1 analog noise generator generating a white noise output signal between DC and 100 kHz. This signal is fed to the LUCIDAC's first analog input port (and the oscilloscope on top).

The LUCIDAC is programmed using the Python package `pybrid` as shown in the following listing:

[1] [Evans 2013] is a great introduction to SDEs. The previous application note as well as this one have been inspired by [Huang et al. 2018].

[2] See [Gronbach et al. 2012, pp. 110 ff.].

[3] See `https://analogparadigm.com/downloads/alpaca_50.pdf`, retrieved 10.12.2025.

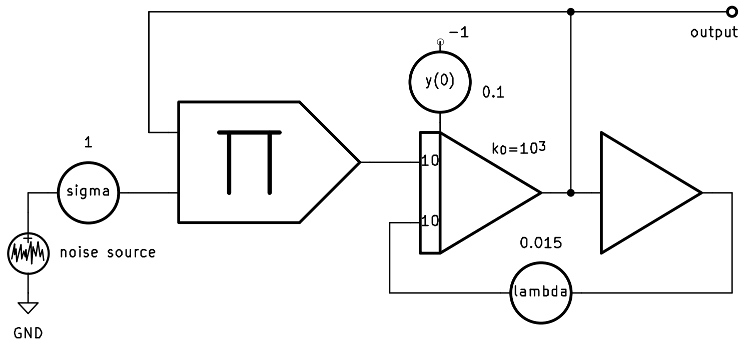

Figure 1: Basic implementation of equation (1)

```
from pybrid.lucidac.lucipy import Circuit, LUCIDAC, time_series
import matplotlib.pyplot as plt
import numpy as np
import json

iters = 1000                                # Parameters
pic = 0.075
plambda = 0.0015
mul_to_int = 1.5
op_secs     = 0.008
sample_rate = 10_000

c    = Circuit()                            # circuit definition
mul0 = c.mul()
W = c.analog_io(id=0)                       # Noise input
c.connect(W, mul0.b)
itor0 = c.int(ic=-pic)

```

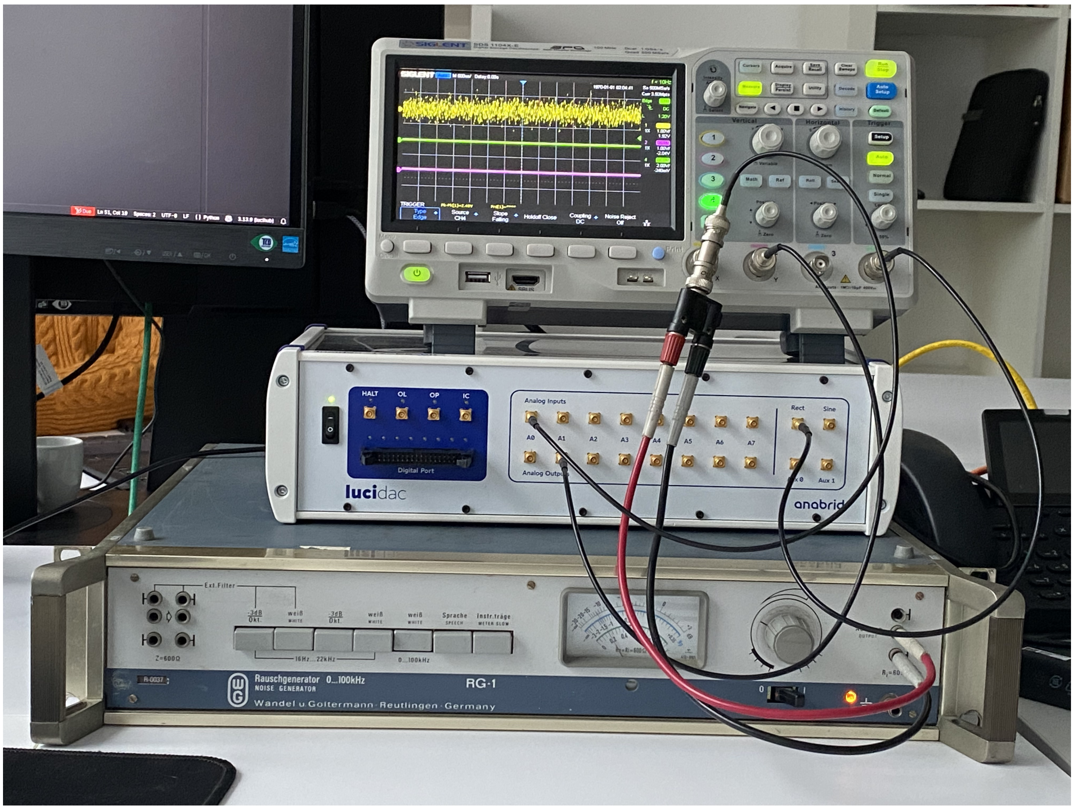

Figure 2: LUCIDAC setup with external analog noise source

```
c.connect(mul0.out, itor0,  weight=mul_to_int)
c.connect(itor0,    itor0,  weight=-plambda)
c.connect(itor0,    mul0.a, weight=-1.0)
c.connect(itor0,    W,      weight=1.0)
c.measure(itor0)                        # Read from itor0

luci     = LUCIDAC()
luci.set_circuit(c)                     # Assign circuit
```

```python

last_samples = []                          # run and capture data
for ctr in range(iters):
  luci.set_daq(num_channels=1, sample_rate=sample_rate)
  luci.set_run(ic_time = 1_000, op_time=int(op_secs * 1_000_000_000))
  try:                                     # Perform a single run
    run = luci.run()                       # We only need the last sample:
    last_sample = list(run.data.items())[0][1][-1]
    last_samples.append(last_sample)
    print(last_sample)
  except:
    print(f"Skipping run {ctr}...")

with open("samples.dat", 'w') as f:
  for sample in last_samples:
    f.write(str(sample) + "\n")

bin_size = 0.05                            # Generate a bin plot
filtered_samples = [s for s in last_samples if 0 <= s <= 1]
bins = np.arange(0, 1 + bin_size, bin_size)
counts, bin_edges = np.histogram(filtered_samples, bins=bins)

plt.figure(figsize=(10, 6))                # Plot bin sizes
plt.bar(bin_edges[:-1], counts, width=bin_size, align='edge', edgecolor='black')
plt.xlabel('Value Range')
plt.ylabel('Count')
plt.title('Bin Plot of Last Samples (0 to 1)')
plt.grid(axis='y', alpha=0.75)
plt.show()
```

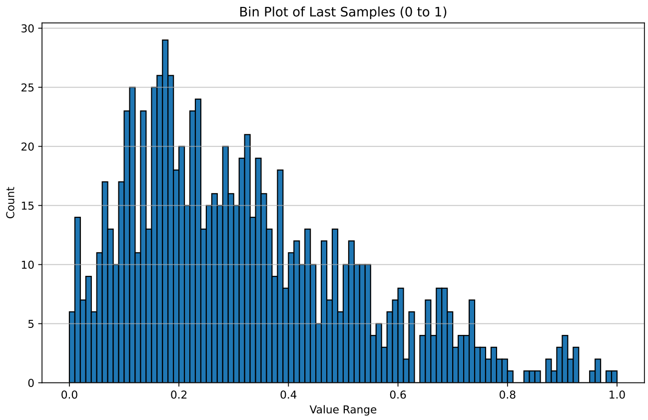

Figure 3: Typical bin-plot

3 Results

Figure 3 shows a typical bin-plot generated by this setup.

Happy analog computing

References

[Evans 2013] Lawrence C. Evans, *An Introduction to Stochastic Differential Equations*, American Mathematical Society, 2013

[Gronbach et al. 2012] Thomas Müller-Gronbach, Erich Nowak, Klaus Ritter, *Monte Carlo-Algorithmen*, Springer Verlag, 2012

[Huang et al. 2018] Yipeng Huang, Ning Guo, Simha Sethumadhavan, Mingoo Seok, Yannis Tsividis, "A Case Study in Analog Co-Processing for Solving Stochastic Differential Equations", 23rd DSP 2018, Shanghai, China