# Temporal Logic Addendum

Jeremy Lau

May 2026

## 1 Introduction

This document provides additional background and details on the implementation of delay-space arithmetic, as published in [2]. A GitHub companion repository [1] provides NumPy implementations of all operations described in this document.

## 2 Importance-space and delay-space

In *importance-space*, the importance of a number $x$ is just $x$, the magnitude of the number itself, so larger numbers have more importance. Importance-space is the "normal" number space that we use every day.

In *delay-space*, the importance of a number $y'$ is $\mathrm{e}^{-y'}$, so *smaller* numbers have more importance. We denote delay-space numbers with a prime symbol ($x'$, $y'$, ...), and importance-space numbers without a prime symbol ($x$, $y$, ...).

We can reversibly map between importance- and delay-space. A delay-space number $y'$ maps to an importance-space number $y$ with:

$$y = \mathrm{e}^{-y'} \tag{1}$$

And an importance-space number $x$ maps to a delay-space number $x'$ with:

$$x' = -\log x \tag{2}$$

Figure 1 shows how importance-space maps to delay-space. Consider how some key numbers in importance space map to delay-space, using Equation 2:

$$\begin{aligned} \lim_{x \to 0} -\log x &= \infty \\ -\log 1 &= 0 \\ -\log \infty &= -\infty \end{aligned}$$

Spend some time studying Figure 1. Understanding this figure is the key to understanding how things work in delay-space. For example:

1. The range $[0, 1)$ in importance-space maps to positive numbers $(\infty, 0)$ in delay-space.
2. The range $(1, \infty)$ in importance-space maps to negative numbers $(0, -\infty)$ in delay-space.
3. Directions are reversed in delay-space. Large positive values in importance-space map to large negative values in delay-space.
4. Delay-space is logarithmic. The blue line does go to $\infty$ and $-\infty$ on the vertical axis, but it gets there *extremely* slowly.
5. Negative importance-space numbers *cannot be mapped* to delay-space, because the log of a negative number is undefined. In Figure 1, the blue line never crosses the vertical axis. Section 3 explains how we work around this limitation.

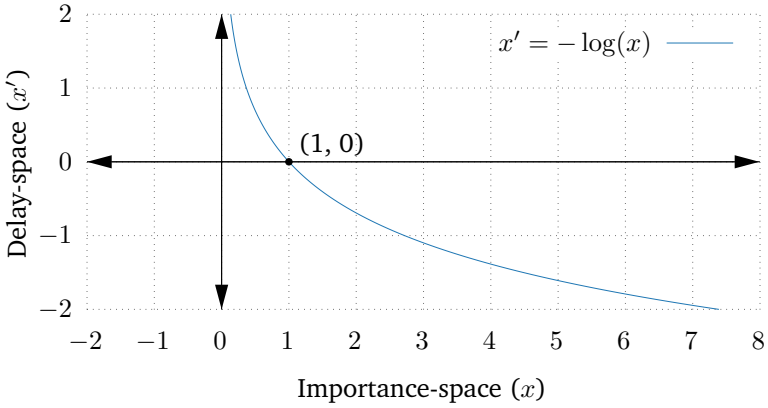


Figure 1: Mapping importance-space to delay-space: $x' = -\log(x)$. Importance-space is on the horizontal axis ($x$), and delay-space is on the vertical axis ($x'$).

# 3 Mapping negative importance-space numbers to delay-space

Negative importance-space numbers can not be directly mapped to delay-space, see Figure 1. We work around this limitation by converting signed importance-space numbers to pairs of unsigned importance-space numbers. These pairs of non-negative numbers can then be directly mapped to delay-space with Equation 2.

Any signed importance-space number $x$ can be represented as a pair of unsigned importance-space numbers $\langle x_{pos}, x_{neg} \rangle$:

$$x_{pos} = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases} \tag{3}$$

$$x_{neg} = \begin{cases} -x & x < 0 \\ 0 & x \geq 0 \end{cases} \tag{4}$$

Some examples: the value $2$ becomes the pair $\langle pos = 2, neg = 0 \rangle$, $0$ becomes $\langle pos = 0, neg = 0 \rangle$, and $-2$ becomes $\langle pos = 0, neg = 2 \rangle$.

Conceptually, $x_{pos}$ represents the magnitude of the positive component of $x$, and $x_{neg}$ represents the magnitude of the negative component of $x$. Only one of these components may be active, with nonzero value, corresponding to the sign of $x$. The inactive component has zero value. In the special case where $x = 0$, both components are zero.

This split $\langle x_{pos}, x_{neg} \rangle$ representation has three key properties:

1. $x_{pos}$ and $x_{neg}$ are unsigned by construction, so they can each be mapped from importance-space to delay-space with Equation 2.

2. The transformation from $x$ to $\langle x_{pos}, x_{neg} \rangle$ is reversible, because $x = x_{pos} - x_{neg}$.

3. There is exactly one $\langle x_{pos}, x_{neg} \rangle$ pair that represents a number $x$, because at most one component may be active, and inactive components must be zero. So even though $1 = 3 - 2$, $\langle pos = 3, neg = 2 \rangle$ is *not* a valid representation of $1$, because at least one of $pos$ and $neg$ must be zero. The *only* valid representation of $1$ is $\langle pos = 1, neg = 0 \rangle$. Combined with property 2, this means that the mapping between $x$ and $\langle x_{pos}, x_{neg} \rangle$ is both one-to-one and onto.

Now we can map any importance-space number $x$ to a pair of delay-space numbers $\langle x'_{pos}, x'_{neg} \rangle$ by com-

bining Equations 2, 3, and 4:

$$x'_{pos} = \begin{cases} \infty & x \leq 0 \\ -\log(x) & x > 0 \end{cases} \tag{5}$$

$$x'_{neg} = \begin{cases} -\log(-x) & x < 0 \\ \infty & x \geq 0 \end{cases} \tag{6}$$

The names $pos$ and $neg$ refer to positive and negative values *only in importance-space*. In delay-space, $x'_{pos}$ can be negative, which can be confusing. Recall that the range $[1, \infty)$ in importance-space maps to the range $[0, -\infty)$ in delay-space, see Figure 1.

# 4 Unsigned delay-space arithmetic

We first define limited versions of delay-space arithmetic operators that only work with unsigned inputs and outputs. We will build full versions of these arithmetic operators in Section 5. The full versions are built using these limited versions, so it helps to start with these definitions.

## 4.1 Unsigned delay-space multiplication ($\mathrm{multiply}'$) and division ($\mathrm{divide}'$)

In delay-space, $\mathrm{multiply}'(x', y')$ must be equivalent to $x \times y$ in importance-space. We derive unsigned multiplication by mapping importance-space multiplication to delay-space with Equation 2:

$$\begin{aligned} x' &= -\log x \\ y' &= -\log y \end{aligned}$$

$$\begin{aligned} \mathrm{multiply}'(x', y') &= -\log(x \times y) \\ &= -\log x + -\log y \\ &= x' + y' \end{aligned} \tag{7}$$

The last step just substitutes the definitions of $x'$ and $y'$.

Equation 7 shows that unsigned delay-space multiplication ($\mathrm{multiply}'$) is implemented with importance-space addition ($+$).

Unsigned delay-space division can be derived very similarly: $\mathrm{divide}'$ is implemented with importance-space subtraction $\mathrm{divide}'(x', y') = x' - y'$.

## 4.2 Unsigned delay-space addition ($\mathrm{nLSE}$)

In delay-space, $\mathrm{nLSE}(x', y')$ must be equivalent to $x+y$ in importance-space. We derive unsigned delay-space addition by following the same procedure:

$$\begin{aligned} \mathrm{nLSE}(x', y') &= -\log(x + y) \\ &= -\log(\mathrm{e}^{\log x} + \mathrm{e}^{\log y}) \\ &= -\log(\mathrm{e}^{-x'} + \mathrm{e}^{-y'}) \end{aligned} \tag{8}$$

This simplifies by substituting $x$ with $\mathrm{e}^{\log x}$, and $y$ with $\mathrm{e}^{\log y}$, then substituting the definitions of $x'$ and $y'$. Equation 8 shows that unsigned delay-space addition ($\mathrm{nLSE}$) is implemented with importance-space negative log-sum-exp.

### 4.3 Unsigned delay-space subtraction ($\mathrm{nLDE}$)

Subtraction in delay-space $\mathrm{nLDE}(x', y')$ can be derived similarly:

$$\begin{aligned}\mathrm{nLDE}(x', y') &= -\log(x - y) \\ &= -\log(\mathrm{e}^{\log x} - \mathrm{e}^{\log y}) \\ &= -\log(\mathrm{e}^{-x'} - \mathrm{e}^{-y'})\end{aligned} \tag{9}$$

Equation 9 shows that unsigned delay-space subtraction ($\mathrm{nLDE}$) is implemented with importance-space negative log-difference-exp. This limited version of delay-space subtraction is only defined when $x' \leq y'$.

## 5 Signed delay-space arithmetic

This section builds full implementations of delay-space arithmetic operators that work with negative importance-space numbers. These full implementations use the limited definitions from Section 4. This section only covers signed multiplication ($\otimes$) signed addition ($\oplus$), and signed subtraction ($\ominus$). Signed division ($\oslash$) is possible, but is not explained in this document.

### 5.1 Signed delay-space multiplication ($\otimes$)

We remove the non-negative constraint from $x$ and $y$ for multiplication, by representing them with a pair of unsigned components $\langle x_{pos}, x_{neg}\rangle$, $\langle y_{pos}, y_{neg}\rangle$, as shown in Section 3. These unsigned components have corresponding delay-space values $\langle x'_{pos}, x'_{neg}\rangle$ and $\langle y'_{pos}, y'_{neg}\rangle$, using Equation 5 and Equation 6.

We wish to compute the delay-space product

$$\langle p'_{pos}, p'_{neg}\rangle = \langle x'_{pos}, x'_{neg}\rangle \otimes \langle y'_{pos}, y'_{neg}\rangle$$

which must be equivalent to

$$p = x \times y$$

in importance-space.

Using the importance-space split representation, $p = x \times y$ can be expanded to:

$$\begin{aligned}p &= \langle x_{pos}, x_{neg}\rangle \times \langle y_{pos}, y_{neg}\rangle \\ p &= (x_{pos} - x_{neg}) \times (y_{pos} - y_{neg}) \\ p &= (x_{pos} \times y_{pos}) - (x_{pos} \times y_{neg}) - (x_{neg} \times y_{pos}) + (x_{neg} \times y_{neg})\end{aligned}$$

These binomial expansion terms can be grouped into positive and negative components. Because all four products of $x_{pos} \times y_{pos}$, $x_{pos} \times y_{neg}$, $x_{neg} \times y_{pos}$, and $x_{neg} \times y_{neg}$ must be non-negative, those terms with a positive sign in front can only contribute to $p_{pos}$, and those with negative signs can only contribute to $p_{neg}$:

$$\begin{aligned}p_{pos} &= x_{pos} \times y_{pos} + x_{neg} \times y_{neg} \\ p_{neg} &= x_{pos} \times y_{neg} + x_{neg} \times y_{pos}\end{aligned}$$

At most one of these binomial expansion terms may be nonzero, because at most one of $x_{pos}$ and $x_{neg}$ may be nonzero, and at most one of $y_{pos}$ and $y_{neg}$ may be nonzero. This lets us replace the importance-space addition with $\max$, because one of the inputs to each addition must be zero:

$$\begin{aligned}p_{pos} &= \max(x_{pos} \times y_{pos}, x_{neg} \times y_{neg}) \\ p_{neg} &= \max(x_{pos} \times y_{neg}, x_{neg} \times y_{pos})\end{aligned}$$

These equations can be converted to delay-space, using the unsigned $\mathrm{multiply}'$ operator introduced in Section 4.1, because the inputs to each multiplication are non-negative:

$$\begin{aligned}p'_{pos} &= \max'(\mathrm{multiply}'(x'_{pos}, y'_{pos}), \mathrm{multiply}'(x'_{neg}, y'_{neg})) \\ p'_{neg} &= \max'(\mathrm{multiply}'(x'_{pos}, y'_{neg}), \mathrm{multiply}'(x'_{neg}, y'_{pos}))\end{aligned}$$

Finally, $\max'$ is implemented by $\min$, because larger values in importance-space correspond to smaller values in delay-space. By expanding Equation 7, we can define general delay-space multiplication $\langle x'_{pos}, x'_{neg}\rangle \otimes \langle y'_{pos}, y'_{neg}\rangle$ as:

$$p'_{pos} = \min(x'_{pos} + y'_{pos}, x'_{neg} + y'_{neg}) \tag{10}$$
$$p'_{neg} = \min(x'_{pos} + y'_{neg}, x'_{neg} + y'_{pos}) \tag{11}$$

## 5.2 Signed delay-space addition ($\oplus$)

We wish to add two delay-space numbers $x' \oplus y' = s'$, which correspond to importance-space numbers $x$ and $y$ with sum $s$. $x$, $y$, and $s$ may be positive, negative, or zero. If we convert $x$ and $y$ to our split $\langle pos, neg\rangle$ pair representation, we see that:

$$\begin{aligned} x &= x_{pos} - x_{neg} \\ y &= y_{pos} - y_{neg} \\ s &= x + y \\ s &= (x_{pos} - x_{neg}) + (y_{pos} - y_{neg}) \\ s &= (x_{pos} + y_{pos}) - (x_{neg} + y_{neg}) \end{aligned} \tag{12}$$

In Equation 12, the inputs to both additions are unsigned, so we can implement these additions in delay-space with $\mathrm{nLSE}$ from Section 4.2.

In Equation 12, the inputs to subtraction, $(x_{pos} + y_{pos})$ and $(x_{neg} + y_{neg})$, are both unsigned, so we can implement this subtraction in delay-space with $\mathrm{nLDE}$ from Section 4.3, *if* we can ensure the difference is also unsigned. That can be done by defining the difference piecewise, and only subtracting when $x > y$. In importance-space, this looks like:

$$\mathrm{nonneg\_sub}(x, y)_{pos} = \begin{cases} 0 & x \le y \\ x - y & x > y \end{cases}$$
$$\mathrm{nonneg\_sub}(x, y)_{neg} = \begin{cases} y - x & x < y \\ 0 & x \ge y \end{cases}$$

This definition of non-negative subtraction ($\mathrm{nonneg\_sub}$) can be mapped to delay-space:

$$\mathrm{nonneg\_sub}'(x', y')_{pos} = \begin{cases} \infty & x' \ge y' \\ \mathrm{nLDE}(x', y') & x' < y' \end{cases} \tag{13}$$

$$\mathrm{nonneg\_sub}'(x', y')_{neg} = \begin{cases} \mathrm{nLDE}(y', x') & x' > y' \\ \infty & x' \le y' \end{cases} \tag{14}$$

With Equations 13 and 14, we can convert Equation 12 from importance-space to delay-space, resulting in a definition of general delay-space addition:

$$\langle x'_{pos}, x'_{neg}\rangle \oplus \langle y'_{pos}, y'_{neg}\rangle = \mathrm{nonneg\_sub}'(\mathrm{nLSE}(x'_{pos}, y'_{pos}), \mathrm{nLSE}(x'_{neg}, y'_{neg})) \tag{15}$$

## 5.3 Signed delay-space subtraction ($\ominus$)

We can negate a value by simply swapping $\langle pos, neg\rangle$ components, both in importance-space and delay-space. So in importance-space, $n = -x$ becomes:

$$\begin{aligned} n_{pos} &= x_{neg} \\ n_{neg} &= x_{pos} \end{aligned}$$

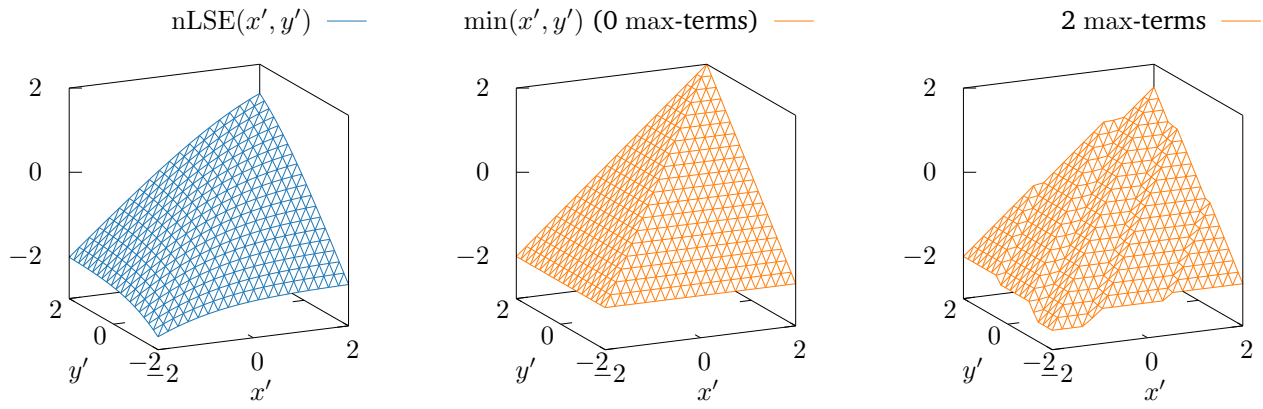


Figure 2: nLSE and two approximations. Additional max-terms improve the approximation. The two right figures approximate the left figure, and the rightmost figure is a better approximation.

And in delay-space, $n' = \ominus x'$ becomes:

$$n'_{pos} = x'_{neg} \tag{16}$$
$$n'_{neg} = x'_{pos} \tag{17}$$

With the ability to negate in delay-space, we can define general delay-space subtraction $\langle x'_{pos}, x'_{neg}\rangle \ominus \langle y'_{pos}, y'_{neg}\rangle$ by composing delay-space addition and delay-space negation:

$$\langle x'_{pos}, x'_{neg}\rangle \ominus \langle y'_{pos}, y'_{neg}\rangle = \langle x'_{pos}, x'_{neg}\rangle \oplus (\ominus\langle y'_{pos}, y'_{neg}\rangle)) \tag{18}$$

# 6 Approximating unsigned delay-space arithmetic

This section describes how we approximate nLSE (Section 4.2) and nLDE (Section 4.3). These approximations are drop-in replacements for nLSE and nLDE, so the approximations can be directly used with the signed delay-space addition equations presented in Section 5.2.

## 6.1 Approximating nLSE

We approximate nLSE with equations of the form:

$$\begin{aligned} \textit{smaller} &= \min(x', y') \\ \textit{larger} &= \max(x', y') \\ \mathrm{nLSE}(x', y') &\approx \min(\textit{smaller}, \max(\textit{larger} + C_0, \textit{smaller} + D_0), \max(\textit{larger} + C_1, \textit{smaller} + D_1), \dots) \end{aligned} \tag{19}$$

These equations can have an arbitrary number of max-terms, where each max-term has the form $\max(\textit{larger} + C_i, \textit{smaller} + D_i)$. We use a solver to find optimal values for the constants $C_i$ and $D_i$, for various numbers of max-terms. Equations with more max-terms produce better approximations, but require more computation. Table 1 provides sample values for several approximations with different numbers of max-terms.

Addition is commutative ($a + b = b + a$). Our nLSE approximation takes advantage of this property by optimizing max-terms for only the $x' \le y'$ case. When $x' > y'$, the *smaller* and *larger* variables in Equation 19 will swap them.

Figure 2 plots nLSE and two approximations.

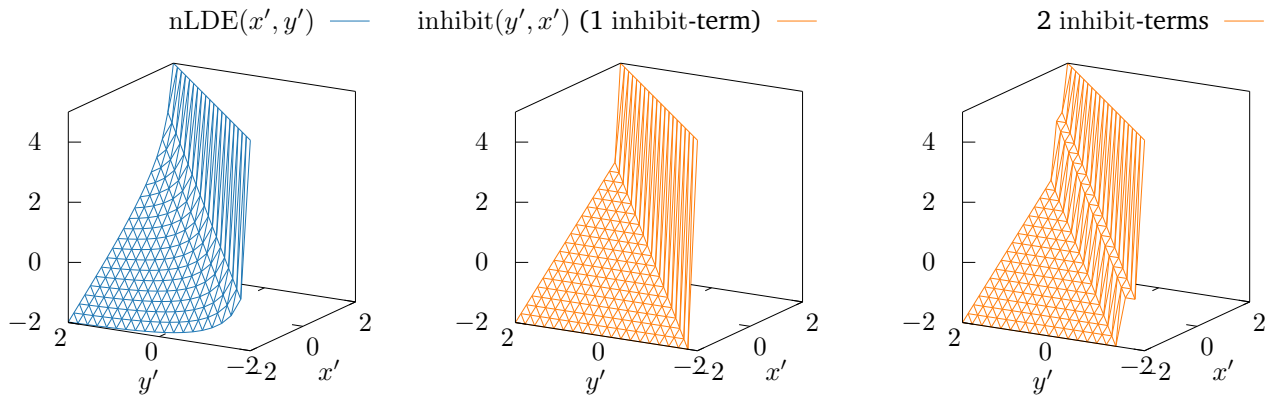


Figure 3: nLDE and two approximations. Additional inhibit-terms improve the approximation. The two right figures approximate the left figure, and the rightmost figure is a better approximation.

### 6.2 Approximating nLDE

We approximate nLDE with equations of the form:

$$\text{nLDE}(x', y') \approx \min(\text{inhibit}(y' + E_0, x' + F_0), \text{inhibit}(y' + E_1, x' + F_1), \dots) \tag{20}$$

Where inhibit is defined as:

$$\text{inhibit}(i, d) = \begin{cases} \infty & i < d \\ d & i \geq d \end{cases} \tag{21}$$

As in Section 6.1, these approximations can have an arbitrary number of terms, where each inhibit-term has the form $\text{inhibit}(y' + E_i, x' + F_i)$. More inhibit-terms result in better approximations, but are more expensive to compute. We use a solver to find optimal values of $E_i$ and $F_i$, and present sample constant values in Table 2.

nLDE is only defined when $x' \leq y'$. Attempting to evaluate nLDE when $x' > y'$ takes the log of a negative number, which typically results in not-a-number (NaN). When $x' > y'$, our nLDE approximation has different behavior than nLDE: the approximation evaluates to $\infty$ instead of NaN. This difference in behavior does not matter in practice, because we should not evaluate nLDE when $x' > y'$. See how Equations 13 and 14 apply this constraint.

Figure 3 plots nLDE and two approximations.

## References

[1] Temporal logic addendum GitHub repository. `https://github.com/UCSBarchlab/delay_space`.

[2] Rhys Gretsch, Peiyang Song, Advait Madhavan, Jeremy Lau, and Timothy Sherwood. Energy efficient convolutions with temporal arithmetic. In *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, April 2024.

| Terms | $\mathrm{nLSE}$ Constants |
|---|---|
| 1 | $C = [-1.6340]$<br>$D = [-0.4386]$ |
| 2 | $C = [-1.2103, -2.5141]$<br>$D = [-0.5396, -0.1691]$ |
| 3 | $C = [-1.0503, -1.7815, -3.1068]$<br>$D = [-0.5832, -0.2782, -0.0909]$ |
| 4 | $C = [-0.9659, -1.4705, -2.2238, -3.5473]$<br>$D = [-0.6076, -0.3507, -0.1717, -0.0575]$ |
| 5 | $C = [-0.9137, -1.2974, -1.8194, -2.5803, -3.8924]$<br>$D = [-0.6232, -0.4021, -0.2364, -0.1175, -0.0401]$ |
| 6 | $C = [-0.8782, -1.1870, -1.5842, -2.1145, -2.8765, -4.1719]$<br>$D = [-0.6340, -0.4402, -0.2880, -0.1712, -0.0861, -0.0300]$ |
| 7 | $C = [-0.8525, -1.1104, -1.4299, -1.8347, -2.3685, -3.1278, -4.4035]$<br>$D = [-0.6419, -0.4697, -0.3296, -0.2175, -0.1304, -0.0662, -0.0235]$ |
| 8 | $C = [-0.8330, -1.0543, -1.3209, -1.6469, -2.0557, -2.5902, -3.3446, -4.5986]$<br>$D = [-0.6480, -0.4930, -0.3638, -0.2572, -0.1709, -0.1031, -0.0528, -0.0190]$ |
| 9 | $C = [-0.8178, -1.0114, -1.2398, -1.5120, -1.8419, -2.2525, -2.7861, -3.5341, -4.7652]$<br>$D = [-0.6528, -0.5120, -0.3923, -0.2912, -0.2070, -0.1382, -0.0840, -0.0434, -0.0159]$ |
| 10 | $C = [-0.8055, -0.9775, -1.1771, -1.4103, -1.6861, -2.0183, -2.4294, -2.9608, -3.7015, -4.9089]$<br>$D = [-0.6566, -0.5277, -0.4163, -0.3206, -0.2391, -0.1707, -0.1145, -0.0699, -0.0364, -0.0136]$ |

Table 1: Sample $\mathrm{nLSE}$ constants with varying numbers of $\max$-terms.

| Terms | $\mathrm{nLDE}$ Constants |
|---|---|
| 1 | $E = [-0.0000]$<br>$F = [-0.0000]$ |
| 2 | $E = [1.6632, -0.6004]$<br>$F = [1.6632, 0.0000]$ |
| 3 | $E = [2.4521, 0.3351, -1.3880]$<br>$F = [2.4521, 0.5838, 0.0000]$ |
| 4 | $E = [2.9843, 0.9382, -0.3782, -1.9405]$<br>$F = [2.9843, 1.0793, 0.3129, 0.0000]$ |
| 5 | $E = [3.3973, 1.3900, 0.2163, -0.8740, -2.3663]$<br>$F = [3.3973, 1.4816, 0.6413, 0.1978, 0.0000]$ |
| 6 | $E = [3.7357, 1.7539, 0.6498, -0.2692, -1.2593, -2.7117]$<br>$F = [3.7357, 1.8183, 0.9401, 0.4316, 0.1374, 0.0000]$ |
| 7 | $E = [4.0225, 2.0593, 0.9955, 0.1619, -0.6409, -1.5755, -3.0010]$<br>$F = [4.0225, 2.1072, 1.2072, 0.6612, 0.3127, 0.1015, 0.0000]$ |
| 8 | $E = [4.2715, 2.3227, 1.2847, 0.5011, -0.2065, -0.9442, -1.8439, -3.2488]$<br>$F = [4.2715, 2.3598, 1.4463, 0.8767, 0.4944, 0.2381, 0.0784, 0.0000]$ |
| 9 | $E = [4.4914, 2.5545, 1.5344, 0.7829, 0.1316, -0.5048, -1.2014, -2.0770, -3.4646]$<br>$F = [4.4914, 2.5841, 1.6619, 1.0765, 0.6717, 0.3856, 0.1880, 0.0627, 0.0000]$ |
| 10 | $E = [4.6884, 2.7616, 1.7543, 1.0251, 0.4103, -0.1654, -0.7564, -1.4250, -2.2826, -3.6549]$<br>$F = [4.6884, 2.7858, 1.8576, 1.2612, 0.8406, 0.5338, 0.3101, 0.1526, 0.0514, 0.0000]$ |

Table 2: Sample $\mathrm{nLDE}$ constants with varying numbers of $\mathrm{inhibit}$-terms.