A Novel Transfer Function for Continuous Interpolation between Summation and Multiplication in Neural Networks

Wiebke Köpp KOEPP@IN.TUM.DE
Technische Universität München, Department of Informatics

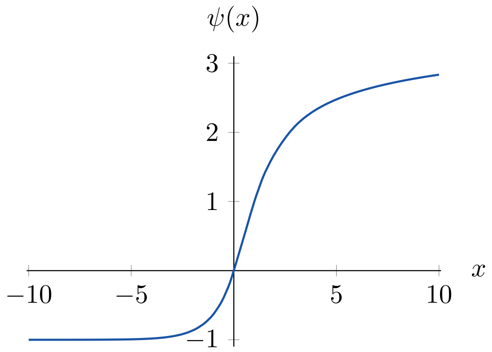

Figure 1. A continuously differentiable solution $\psi(x)$ for Abel's equation (**??**) with $f(x) = \exp(x)$.

Abstract

Abstract

1. Fractional Iterates of the Exponential Function

1.1. Abel's Functional Equation

Assuming $\psi(x) = x$ for $0 \leq x < 1$, we can find a solution for ψ which is valid on all of $\mathbb{R}$. Since we have $\log(x) < x$ for any $x \geq 1$, multiple applications of (**??**) will, at some point, lead to an argument for ψ that falls into the interval where we assume ψ to be linear. In case $x < 0$, we can use $\psi(x) = \psi(\exp(x)) - 1$ to reach the desired interval, since $0 < \exp(x) < 1$ for $x < 0$. In combination, this leads to a piece-wise defined solution for ψ,

$$\psi(x) = log^{(k)}(x) + k \tag{1a}$$

$$\text{with } k \in \mathbb{N} \cup \{-1, 0\} : 0 \leq \log^k(x) < 1\,. \tag{1b}$$

The function is displayed in Fig. 1.

2. Implementation Details

This section first details how the real- and complex fractional exponential functions are implemented in an efficient manner and then discusses their integration in Theano. Considerations about numerical accuracy and improvements in computational efficiency are covered thereafter.

Algorithm 1 Computation of $\psi(x)$.

```
if x < 0 then return exp(x) − 1
else
    k ← 0
    while x > 1 and k < k_max(= 5) do
        x ← log(x); k ← k + 1
    end while
    return x + k
end if
```

r_z	CPU e_{rel} $[10^{-5}]$	CPU t [ms]	GPU e_{rel} $[10^{-5}]$	GPU t [ms]	M [MB]
0.1	27.45	269.13	31.05	9.29	0.62
0.05	6.757	274.57	9.698	10.38	2.45
0.01	0.278	388.48	1.313	15.80	60.91
0.0075	0.160	421.63	0.987	15.72	108.20
0.005	0.080	478.90	0.685	16.56	243.46
0.0035	0.050	490.75	0.541	16.85	496.60

r_z	r_n	CPU e_{rel} $[10^{-3}]$	CPU t [ms]	GPU e_{rel} $[10^{-3}]$	GPU t [ms]	M [MB]
0.1	0.1	9.865	63.83	9.877	6.58	12.46
0.1	0.05	2.487	123.31	2.497	7.59	24.32
0.05	0.1	9.790	168.43	9.799	7.65	49.52
0.05	0.05	2.456	180.06	2.465	8.54	96.70
0.1	0.01	0.116	187.69	0.138	8.00	119.25
0.05	0.025	0.619	190.78	0.628	8.55	191.03
0.075	0.01	0.101	200.08	0.129	8.72	210.69
0.075	0.005	0.003	212.79	0.005	8.12	420.32

Table 1. Results for the interpolation of $\exp^{(n)}$ with method A (left) and method B (right). r_z and r_n are the sampling resolutions used for choosing the values to be precomputed. t denotes the runtime of computing function values for one million test points and M gives the required memory for the respective setting. All GPU computations have been conducted on a Nvidia Quadro K2200. The used CPU is a Intel Xeon E3-1226 v3 @ 3.30 GHz.

Algorithm 2 Computation of $\psi^{-1}(\psi)$. The value for ψ^{-1} with $\psi < -1$ would mathematically be $-\infty$. Similarly, inputs with $\psi \gtrsim 4.406$ evaluate to values larger than can be represented with single precision floating point numbers. Since infinite values cause problems in gradient-based learning, we restrict ψ^{-1} to the interval $[-10, 10]$. ψ_{min} and ψ_{max} are set as $\psi_{min} = -0.999955$ and $\psi_{max} = 2.83403$, so that $\psi^{-1}(\psi_{min}) = -10$ and $\psi^{-1}(\psi_{max}) = 10$.

if $\psi < \psi_{min}$ **then return** $\psi^{-1}(\psi_{min})$
else if $x > \psi_{max}$ **then return** $\psi^{-1}(\psi_{max})$
else
  $k = \lceil \psi - 1.0 \rceil$
  if $k < 0$ **then**
    return $\log(\psi - k)$
  end if
  $\psi \leftarrow \psi - k$
  while $k > 0$ **do**
    $\psi \leftarrow \exp(\psi)$; $k \leftarrow k - 1$
  end while
  return ψ
end if