Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 55-20 (2022) 529–533

Regularization of the Logarithmic Mean for Robust Numerical Simulation

Niels Weber * **Dirk Zimmer** *

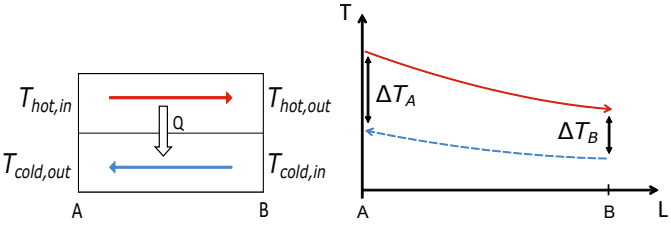
* *German Aerospace Center (DLR), Institute of System Dynamics and Control, Münchener Str. 20, 82234 Weßling (e-mail: {niels.weber,dirk.zimmer}@dlr.de)*

Abstract:
To calculate the driving temperature difference between the hot and the cold side of a heat exchanger, the use of the logarithmic mean temperature difference is common practice. To provide high robustness in complex dynamic system models, a robust formulation of the logarithmic mean (logmean) function becomes vital. As the analytic definition of the logmean function naturally comes along with singularities and limitations for specific input conditions, it is essential to extend and modify it for heat exchanger modeling. This paper proposes how the logmean function can be extended to be valid in all four quadrants of the Cartesian coordinate system and how to bridge the resulting definition gaps. Special focus lies on the robust formulation in such a way that it can be easily handled by numerical solvers. This includes the numerical approximation of the logmean by use of its integral form by implicit ODE solvers with variable step width. Furthermore a way is presented to flatten the naturally steep gradients in the vicinity of the x- and y-axes without manipulating the function in the uncritical regions. All the modifications on the logmean are finally applied in a simple simulation model written in the object-oriented programming language Modelica to examine the robustness of the approach.

Keywords: Logarithmic mean, regularization, heat exchangers, robust modeling, Modelica, thermal systems

1. INTRODUCTION

The modeling and simulation of complex thermal system architectures plays an increasing role in the development of future energy systems. Such systems can represent the thermal management of electric vehicles, aircraft environmental control systems, power plants or any kind of cooling circuit. When dealing with such complex thermal system architectures, the robust formulation of all components is inevitable. At our institute, a lot of effort is put into the robust modeling of complex thermofluid networks (Zimmer et al. (2018) and Zimmer et al. (2021)).



(a) Flow arrangement of counter-flow heat exchanger (b) Temperatures along heat exchanger

Fig. 1. Principle of counter-flow heat exchangers

Heat exchangers are frequently used in such models and hence become a central part in thermal system architectures. In the following, we take a closer look into a common method to obtain the heat flow through a heat exchanger when the inlet and outlet temperatures are given. In general, the heat flow through a heat exchanger can be described by the following equation given for example in Incropera et al. (2007):

$$Q = UA\Delta T \tag{1}$$

with the heat flow rate Q, the overall heat transfer coefficient U and the heat conducting area A. The temperature difference ΔT determines the driving force for the heat flow across the heat exchanger. As the temperature difference between the hot and the cold side varies along the heat exchanger, a mean temperature difference is used. To this end, the logarithmic mean temperature difference ΔT_{LM} is introduced to calculate the heat flow:

$$Q = UA\Delta T_{LM}. \tag{2}$$

Figure 1 gives an overview about the inlet and outlet temperatures of a generic heat exchanger in counter-flow arrangement. The relevant temperature differences are given with $\Delta T_A = T_{hot,in} - T_{cold,out}$ and $\Delta T_B = T_{hot,out} - T_{cold,in}$. Those temperature differences are used to calculate the logarithmic mean temperature difference ΔT_{LM}:

$$\Delta T_{LM} = \frac{\Delta T_A - \Delta T_B}{ln\left(\frac{\Delta T_A}{\Delta T_B}\right)} = \frac{\Delta T_A - \Delta T_B}{ln(\Delta T_A) - ln(\Delta T_B)}. \tag{3}$$

10.1016/j.ifacol.2022.09.149

The closed form of the logarithmic mean (logmean) function reveals several singularities and limitations for heat exchanger modeling. It is only defined for both temperature differences being positive values. In transient zones this condition might temporarily not be fulfilled. This phenomena is called temperature crossing and can appear, when the fluids in the heat exchanger experience sudden temperature changes. The cold fluid therefore can reach its maximum temperature already inside the heat exchanger and not at the outlet. Also the domain for both temperature differences being negative is not covered. Furthermore very steep gradients appear in the vicinity of the x- and y-axes. It is therefore inevitable to apply appropriate modifications to the analytic form of the logarithmic mean function. It turned out as a quite challenging task to modify the function in a way it can be used for robust heat exchanger modeling. So lets have a closer look on the logmean function to understand its behavior and how we have to adjust it to our purpose.

2. REGULARIZING THE LOGARITHMIC MEAN FUNCTION

This chapter describes how we have to extend and adjust the typical formulation of the logmean function (see Carlson (1972)) to be valid for all possible input values that can appear in dynamic simulation models including heat exchangers.

2.1 Analytic form

The analytic formulation of the logmean function $L(x, y)$ is given by:

$$L(x, y) = \frac{y - x}{ln(y) - ln(x)}. \tag{4}$$

It is defined for two non-negative numbers x and y and is set to x if $x = y$:

$$L(x, y) = \begin{cases} x & \text{if } x = y, \\ \dfrac{y - x}{ln(y) - ln(x)} & \text{otherwise.} \end{cases}$$

The special case of $x = y$ is given, when the temperature differences at both ends of the heat exchanger are equal. It is then assumed, that the temperature difference stays constant across the entire length of the heat exchanger.

Figure 2 shows the values of the logmean in all three dimensions. The original definition of the logmean function is therefore only valid in the first quadrant of the Cartesian coordinate system. This would limit its use for robust heat exchanger modeling, because temperature differences of all possible sign-combinations can occur in transient phases of dynamic models. To overcome this limitation we have to expand the function to all four quadrants. This is done by slightly modifying the natural definition as follows:

$$L(x, y) = \begin{cases} 0 & \text{if } x * y < 0, \\ x & \text{if } x = y, \\ \dfrac{y - x}{ln(\frac{y}{x})} & \text{otherwise.} \end{cases}$$

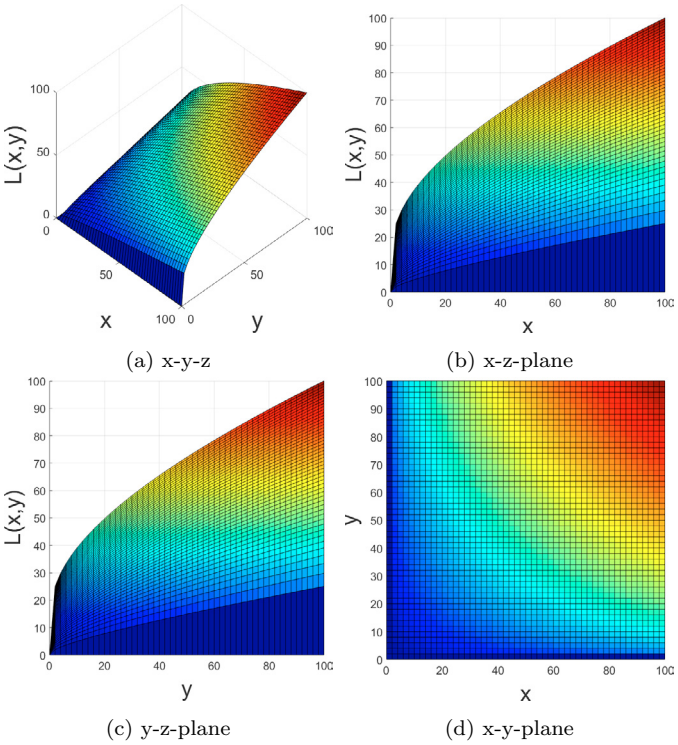


Fig. 2. Validity of the analytic logarithmic mean

We now allow x and y to be negative numbers. This modification mirrors the function to the third quadrant to be valid for the case when x and y are both negative numbers. As we need the logmean being valid in all four quadrants, we still have to find a formulation for the second and fourth quadrant. In those quadrants, the numbers x and y have different signs. In heat exchanger modeling this phenomena is called temperature crossing. A meaningful solution for this region is to set all values to zero. This guarantees us a smooth transition trough zero, when one of the input values (temperature differences) changes its sign. To bridge the definition gap at $x = y$, it is set to x and hence applies in all quadrants.

Now we are able to plot the logmean function in all four quadrants for x and y both ranging from -100 to 100:

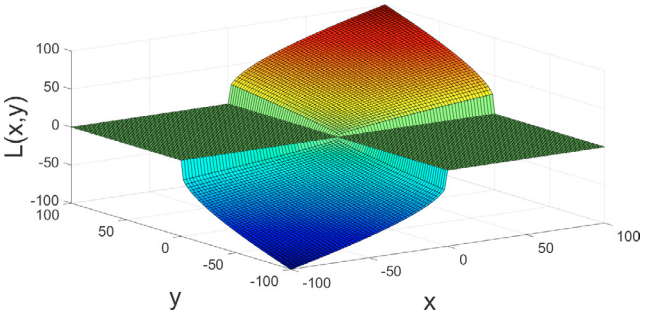


Fig. 3. Surface plot of logmean function in all quadrants

The previous modifications helped to overcome the limitations and singularities that come along with the natural definition of the logmean. We now cover the entire number range and found a solution for the singularity when $x = y$. One might think that the job is done now, but the logmean function offers some additional tricky pitfalls when it has to be implemented for a numerical solver.

2.2 Numerical Integration

The logarithmic mean can also be interpreted as the area under an exponential curve. For numerical solvers it is more feasible to approximate the integral using a quadrature method instead of implementing the closed form of the logarithmic mean function.

By exploiting the integral form, we get rid of the logarithm and end up with a fairly easy integral to approximate in the definite interval [0,1]:

$$L(x,y) = x \int_0^1 \left(\frac{y}{x}\right)^t \mathrm{d}t. \tag{5}$$

For the approximation of the integral we apply the trapezoidal rule. It is a common quadrature method that promises robustness for our applications.

The general form of the trapezoidal rule for non-uniform grid spacing is given by:

$$\int_a^b f(x)dx \approx \sum_{k=1}^{N} \frac{f(x_{k-1}) + f(x_k)}{2} \Delta x_k \tag{6}$$

with grid points x_k and $\Delta x_k = x_k - x_{k-1}$ and the total number of grid points being $N+1$. An increase of grid points leads to a higher accuracy in the approximation of the integral. For better understanding, Figure 4 serves as a visualization of the integral approximation using the trapezoidal rule and the respective grid points $x_0 ... x_N$.

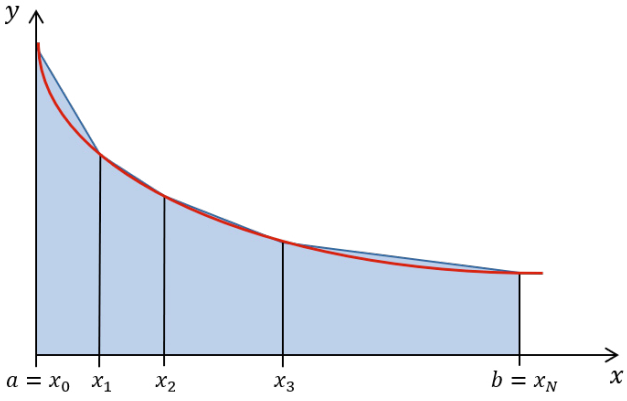


Fig. 4. Sketch of integral approximation using trapezoids

To analyse how close we are to the actual value of the definite integral, we have to conduct an error analysis.

2.3 Error analysis

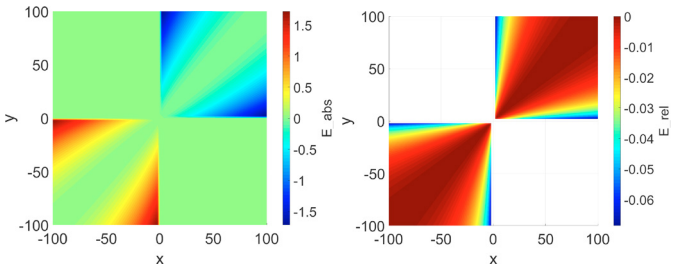
The following error analysis is performed for $N = 4$ and the quadrature method is applied using the grid points $x_0 = 0$, $x_1 = \frac{1}{8}$, $x_2 = \frac{1}{4}$, $x_3 = \frac{1}{2}$ and $x_4 = 1$. With this rather low number of grid points, we are still expecting to be close enough to the solution of the integral.

In general the error E can be estimated by subtracting the value of the integral from the numerical result. For the sake of simplicity we take the numerically approximated form of the logmean L_{approx} and subtract it from the analytical form L_{ana}. Doing so, we can calculate the absolute error $E_{abs}(x,y)$ from:

$$E_{abs}(x,y) = L_{ana}(x,y) - L_{approx}(x,y). \tag{7}$$

Figure 5a shows the absolute error in the x-y plane for x and y both ranging from -100 to 100. It can be observed, that the error increases towards the axes ($x, y \to 0$). The maximum deviation to the analytic solution is < 2. For further analysis, the relative error $E_{rel}(x,y)$ to the analytic form is calculated from:

$$E_{rel}(x,y) = \frac{L_{ana}(x,y) - L_{approx}(x,y)}{L_{ana}(x,y)}. \tag{8}$$



(a) Absolute error (b) Relative error

Fig. 5. Deviation from analytic form

The values of the relative error are given in Figure 5b. Here it should be noted, that the relative error is only given for the first and third quadrant. In the other two quadrants the function is set to zero for both approaches and hence the error is zero. The maximum deviation can be observed near the axes with a maximum value of $E_{rel} < 7\%$. This relative error can be accepted as it appears very locally, when the magnitudes of x and y are very different to each other. In summary it could be shown that a low number of grid points is already sufficient and the resulting error is pretty small. Nevertheless if needed, the accuracy of the approximation could be improved by implying more grid points to the trapezoidal rule.

What is also evident from the error analysis are the steep gradients near the axes. When applying implicit ODE solvers like DASSL (Petzold (1982)), the steep gradients caused a drastic reduction in step-size and a major performance loss. In interaction with other processes even high frequency limit cycle behavior can result. A smoothing of the edges and hence reducing their steepness seemed to be an adequate mean to overcome this unwanted behavior.

2.4 Smoothing

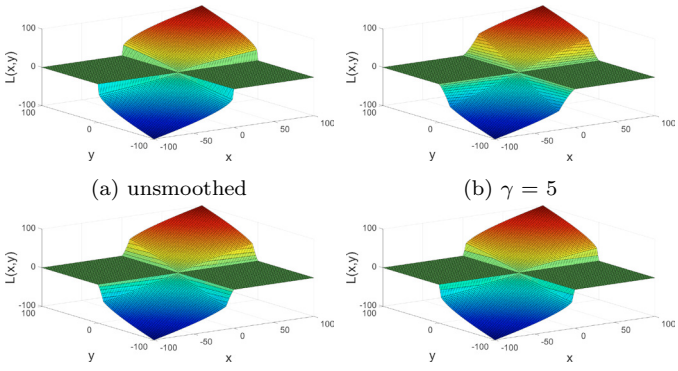
In order to counteract the step-size reduction in such critical areas, an approach to smooth the edges of the logmean function had to be developed. The goal was to locally modify the edges of the function without manipulating the values in the unproblematic regions. To do so, a smoothing factor γ was introduced which constraints the prefactor $\tilde{x}$ of the integral with a maximum slope:

$$\tilde{x} = \begin{cases} \min(x, \gamma y) & \text{if } y > 0, \\ \max(x, \gamma y) & \text{otherwise.} \end{cases}$$

The resulting $\tilde{x}$ can now be placed in the integral form of the logmean (Eq. 5) as follows:

$$L(x,y) = \tilde{x} \int_0^1 \left(\frac{y}{x}\right)^t \mathrm{d}t. \tag{9}$$

The result for different values of the smoothing factor γ is shown in Figure 6. It is clearly visible that the edges of the function near the x- and y-axis can be flattened to a greater or lesser extent by choosing different values for γ.



(a) unsmoothed (b) $\gamma = 5$

(c) $\gamma = 10$ (d) $\gamma = 25$

Fig. 6. Applying different values of smoothing factor γ to original function

When increasing the value of γ, the closer the result gets to the curve without smoothing. Lower values of γ are flattening the edges to a greater extent. This becomes more visible in Figure 7, where the resulting function is plotted along the y-axis..

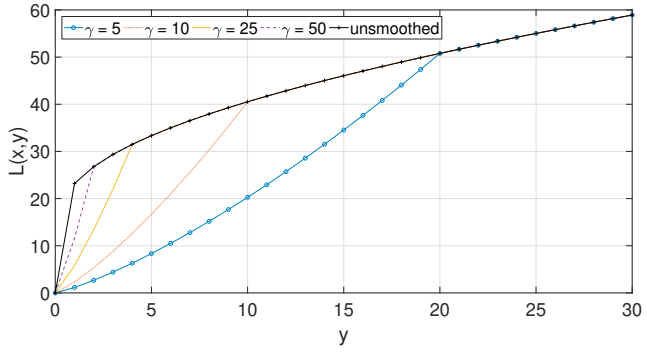


Fig. 7. Impact of different smoothing factors ($x = 100$)

The last remaining regions of the presented formulation of the logmean function that need some more attention are the planes in the second and fourth quadrant. As the value of the logmean is set to zero in those quadrants, no gradient can be found here. For gradient-based methods this will be quite tricky to solve, as the convergence can not be guaranteed. One possible way to prevent from this is to slightly tilt the planes and thus providing a small gradient to support the numerical solver. This is done by adding a fraction α of the arithmetic mean to the logarithmic mean in the following way:

$$L(x,y) = (1-\alpha)L(x,y) + \alpha\frac{x+y}{2}. \tag{10}$$

Consequently, the "tilting-factor" α has to be a number in the range of 0 and 1. Figure 8 depicts the contours of the resulting function for $\alpha = 0.2$. The resulting gradient in the second and fourth quadrant is now clearly visible. It has to be mentioned, that for the sake of visibility, a fairly high value of α is chosen here. Very small gradients in those areas and therefore very small values for α can already be sufficient for numerical solvers. This has to be taken with care and kept in mind for each individual application of this modification.

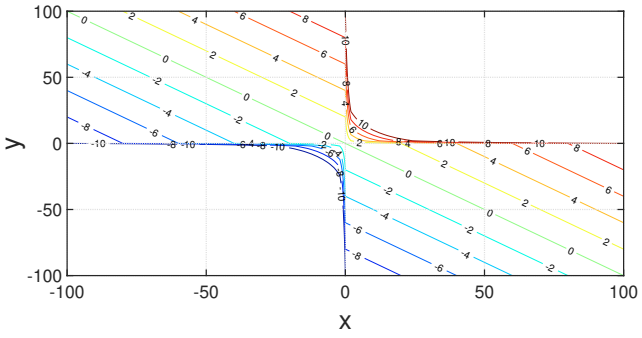


Fig. 8. Contours of logmean with $\alpha = 0.2$

To conclude the paper we plug in the logmean function with all the modifications into a representative model in the modeling language of our choice.

3. IMPLEMENTATION AND TESTING IN MODELICA

For the modeling and simulation of thermal architectures, the object-oriented programming language Modelica is widely used at our institute. Hence we want to give an insight on how we implemented the logmean function in Modelica, whereas the presented modifications are not meant to be Modelica specific. As in this section the focus solely lies on the testing of the logmean function, it is done outside of the frame of a heat exchanger in an isolated test model.

3.1 Modelica Implementation

To examine the behavior of the modified logmean function, a very simple minimal model shall do the job. It is represented by a block that contains the logmean function and the respective inputs x and y and the output $L(x,y)$. The algorithm section of the implementation is given in Listing 1:

Listing 1: Implementation of the logmean

```
algorithm

f := abs(y*x/(x^2+x_norm^2));

//Loop to apply trapezoidal rule
for i in 2:N+1 loop

        dx := grid[i] - grid[i-1];

        L := L + (1/2)*(f^(grid[i-1])+f^(grid[i
            ]))*dx;
end for;
```

```
//Smoothing the edges
if y > 0 then
        x_s :=min(x, gamma*y);
else
        x_s :=max(x, gamma*y);
end if;

L := x_s*L;

//X and y having different signs
if x*y < 0 then
        L := 0;
end if;

//x=y
if y - x == 0 then
        L := x;
end if;

//Tilting
L:= (1-alpha)*L + alpha*(x+y)/2;
```

It has to be denoted here, that array indexing in Modelica starts with 1. For this reason the starting point of the loop for the trapezoidal rule hat to be moved to 2. The array variable `grid` contains the manually defined grid points. Another special case that hasn't been considered in the previous sections yet is given when both inputs x and y are equal to zero. This would result in a division by zero. To prevent the division by zero a common normalization method is applied:

$$\left|\frac{y}{x}\right|_{norm} = \left|\frac{yx}{x^2 + x_{norm}^2}\right| \tag{11}$$

with x_{norm} being a very small number ($x_{norm} = 10^{-5}$).

3.2 Testing

The objective is to cover the whole scope of the logmean definition while retrieving meaningful results for all inputs. To examine whether we can retrieve a meaningful value for the logmean for every possible inputs, we are considering a test case that covers all of the following critical conditions:

- both inputs positive
- both inputs negative
- inputs with opposite sign
- both inputs zero

Figure 9 shows the result of the Modelica model. All possible combinations of x and y are covered with the corresponding input time-table. The model was simulated using the implicit DASSL solver with a tolerance of 10^{-4}. As expected, the solution of the logmean travels smoothly through all conditions and hence shows the desired behavior. For the given model the smoothing factor γ is set to 15 and the tilting-factor α is set to 0.01. This fairly simple test model has been sufficient to examine the implementation of the logmean in Modelica. The presented implementation of the logmean with all its modifications, proved to be robust in more complex system architectures that include several controllers.

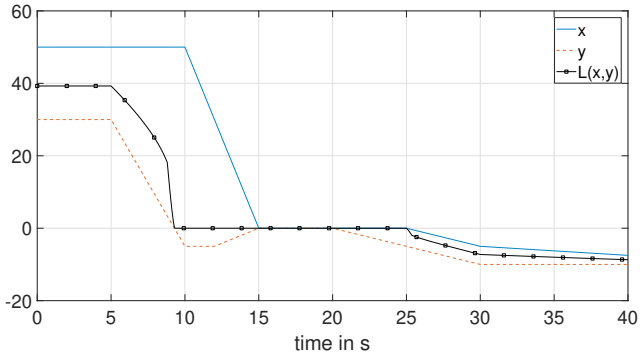


Fig. 9. Result of Modelica model

4. CONCLUSION

In summary it became very clear, that the analytic definition of the logarithmic mean is usually not suitable for robust simulation of heat exchangers especially if they undergo uncommon transients as this may happen in complex architectures. Finding an appropriate formulation of the logmean function turned out to be more challenging than expected. The presented methods to reformulate the logmean function are intended to serve as a toolbox to improve the robustness while maintaining validity. Sacrifices in the validity only have to be made in limited areas that should not affect the stationary result. After all, the point has to be stated that the amount and the extent of the presented modifications of the logmean function can be varied depending on the field of application. Our motivation for putting substantial effort in the regularization of the logmean function were oscillations in heat exchangers that are part of fairly complex system architectures. The resulting reductions in simulation performance made us to revise the implementation of the logmean. For smaller models, for instance only the extension of the logmean to all four quadrants could be sufficient enough. In any case, a simulation engineer using the logarithmic mean should be aware about all the potential regularization that might be needed to make this function applicable in practice.

REFERENCES

Carlson, B.C. (1972). The logarithmic mean. *The American Mathematical Monthly*, 79(6), 615–618. URL `http://www.jstor.org/stable/2317088`.

Incropera, F., Lavine, A., Bergman, T., and DeWitt, D. (2007). *Fundamentals of heat and mass transfer*. Wiley New York.

Petzold, L.R. (1982). A description of dassl: a differential/algebraic system solver.

Zimmer, D., Bender, D., and Pollok, A. (2018). Robust modeling of directed thermofluid flows in complex networks. In *Proceedings of the 2nd Japanese Modelica Conference*, 39–48. Linköping University Press.

Zimmer, D., Weber, N., and Meißner, M. (2021). The dlr thermofluidstream library. In *Modelica Conferences*, 225–234.