`Relleno`: a `SageMath` library for jump expansions and Jacobian matrices of conservation laws and entropy stability analysis

Michael A. Hansen

January 7, 2020

1 Introduction

`Relleno` is a library atop the `SageMath` symbolic manipulation code that facilitates derivation of new techniques for numerically solving conservation laws in scientific computing. Computational simulation of fluid flow across the ranges of high-speed compressible gases, incompressible liquids, and magnetohydrodynamic flows depends on our ability to construct two-point flux functions, which play a large role in the stability, accuracy, and other properties of a numerical method. A simple scheme requiring two-point flux functions is a classical cell-centered finite volume method wherein solution data exists at cell centers while fluxes are needed at cell faces. Favorable properties of a simulation such as stability, accuracy, and preservation of quantities such as kinetic energy and entropy functions translate into requirements on two-point flux functions. `Relleno` facilitates derivation of special flux functions, particularly in the context of entropy stable methods for hyperbolic conservation laws.

As a simple example, consider a one-dimensional hyperbolic PDE,

$$q_t + f(q)_x = 0, \tag{1}$$

which represents the time evolution of a quantity q in space x due to the flux $f(q)$. A simple flux function, which yields the canonical one-dimensional wave equation, is $f(q) = uq$ for a velocity u. Another choice which yields Burgers' equation is to use $f(q) = q^2/2$. Many conservation equations of physical relevance can be cast as systems of hyperbolic PDEs.

When solving Equation (1) numerically, for instance with a cell-centered finite volume method, one needs to compute the flux f at the faces of control volumes despite the fact that q is known only at cell centers. A two-point flux function is simply a numerical approximation, $f^{\mathrm{num}}(q_\ell, q_r)$, to the flux f at a face that involves the two surrounding states, q_ℓ and q_r ('left' and 'right'). The two most 'obvious' choices for computing the face flux are to average q and compute f, or compute f at the cells and average it, as shown below, respectively:

$$f^{\mathrm{num}}(q_\ell, q_r) = f\left(\frac{q_\ell + q_r}{2}\right). \tag{2}$$

$$f^{\mathrm{num}}(q_\ell, q_r) = \frac{f(q_\ell) + f(q_r)}{2}, \tag{3}$$

While the difference between these flux functions may not appear large at first glance, they will perform dramatically differently for problems involving sharp gradients or discontinuities such as shocks (gradient-steepening phenomena).

The problem of choosing a two-point flux function is compounded by the fact that there are infinitely many options. Above we choose an arithmetic average, but in theory any two-point average is valid. Furthermore we can design schemes specifically for particular equation sets. For the wave equation with $f = uq$, we could choose a different average for the velocity and the transported quantity. For example,

$$f^{\text{num}}(q_\ell, q_r) = \overline{u}\,\widehat{q}, \tag{4}$$

where $\overline{u}$ is the arithmetic average and $\widehat{q}$ is the 'logarithmic average,' a specially-defined function that is actually required in certain cases. Further complexity shows up if the transported quantity can be decomposed into a product, for instance $q = \rho q'$. Then we may represent q in the flux with an average the product $\overline{\rho q'}$ or take the product of averages, *i.e.*, $\widehat{\rho}\overline{q'}$.

A key ingredient in deriving two-point flux functions which satisfy certain properties (*e.g.* conservation of functions the state such as kinetic energy or an entropy function) is what we call the 'jump expansion' which is also referred to as 'linearization.' Jump expansion appears naturally in deriving special two-point flux functions and Roe average matrices. Much in the same way as shown above for averages, jump expansions are not unique. Furthermore, computing a jump expansion is tedious and error-prone. The primary purpose of `Relleno` is to automatically compute optimal jump expansions where feasible and facilitate iteration over the options where desired. It is specifically aimed at flux functions for the compressible Euler and Navier-Stokes equations but is a general-purpose library. `Relleno` also facilitates calculus involving complicated systems of equations whose Jacobian matrices, especially those relevant to entropy stability analysis, are awkward to derive with `SageMath` alone.

This writeup is effectively the 'theory manual' of `Relleno`, detailing the mathematics underlying `Relleno`, not its usage or application space. In §2 we discuss expression trees, which are necessary for the recursive jump expansion algorithm in `Relleno`. Section 3 details direct averages (*e.g.*, arithmetic, harmonic) and special indirect averages (*e.g.*, logarithmic) and introduces a handy 'trick' for simplifying averages of powers of quantities (*e.g.*, $\overline{x^2}$). Finally, §4 details jump expansions for fundamental elements of expression trees which are used to compose jump expansions of arbitrarily complex expressions.

2 Expression Trees

2.1 Elements of Trees

We categorize elements of algebraic expressions into auxiliary variables, $\mathcal{A} \subset \mathbb{R}^{n_a}$, constants[1] which are real numbers $\mathbb{R}$ and integers $\mathbb{Z}$, and multiple-input, single-output functions of constants and auxiliary variables, $\mathcal{F} = \{f_i \colon \mathcal{A} \to \mathbb{R}\}$. Expression *trees* are then built up through unary, binary, and n-ary operations, such that only constants and auxiliary variables remain as leaves of the tree. Expression tree decomposition is not unique. For instance, $f^6 = (f^3)^2 = (f^2)^3$ is three distinct tree representations for the same expression.

Unary operations involving only one expression, say, g, include the negative, $\text{neg}(g) = -g$, the reciprocal, $\text{rec}(g) = 1/g$, the natural logarithm $\ln(g)$, exponential $\exp(g)$, arbitrary logarithm of base $r \in \mathbb{R}^+$, $\log_r(g)$, arbitrary exponentiation of a real number $r \in \mathbb{R}$, r^g, trigonometric functions such as $\cos(g)$, $\sin(g)$, etc., and exponentiation of g in the ways enumerated in Table 1. We denote a general unary operation on g as $\text{uop}(g)$.

Binary operations - those involving two expressions g and h - are addition, $g + h = \text{add}(g, h)$, multiplication, $gh = \text{mlt}(g, h)$, and exponentiation, $\text{pow}(g, h) = g^h$. n-ary operations are then simply linked binary operations. The associativity of addition and multiplication means we can

[1]It's entirely possible that this entire document is valid for complex numbers, but we make no guarantees.

Table 1: Exponentiation of an expression g with real numbers $r_+ \in \mathbb{R}^+$ and $r_- \in \mathbb{R}^-$, $|r_-| = r_+$, integers $n_+, k \in \mathbb{Z}^+$ and $n_- \in \mathbb{Z}^-$, $|n_-| = n_+$.

ID	Description	Expression
A	arbitrary power with r_+	g^{r_+}
B	arbitrary power with r_-	$g^{r_-} = \text{rec}(g^{r_+})$
C	even integer power with $n_+ = 2k$	$g^{n_+} = g^{2k} = (g^k)^2$
D	even integer power with $n_- = -2k$	$g^{n_-} = \text{rec}(g^{n_+}) = \text{rec}((g^k)^2)$
E	odd integer power with $n_+ = (2k+1)$	$g^{n_+} = g^{2k}g$
F	odd integer power with $n_- = -(2k+1)$	$g^{n_-} = \text{rec}(g^{n_+}) = \text{rec}(g^{2k}g)$
G	inverse integer power with n_+	g^{1/n_+}
H	inverse integer power with n_-	$g^{1/n_-} = \text{rec}(g^{1/n_+})$

link sub-groups in any way we like. The right associativity of exponentiation, however, is a key distinction - as is the accumulation of exponent expressions into a single product shown below.

$$\begin{aligned} f+g+h+k &= \text{add}(f, \text{add}(g, \text{add}(h,k))), \\ &= \text{add}(\text{add}(\text{add}(f,g),h),k), \\ &= \text{add}(\text{add}(f,g), \text{add}(h,k)), \end{aligned} \tag{5}$$

$$\begin{aligned} fghk &= \text{mlt}(f, \text{mlt}(g, \text{mlt}(h,k))), \\ &= \text{mlt}(\text{mlt}(\text{mlt}(f,g),h),k), \\ &= \text{mlt}(\text{mlt}(f,g), \text{mlt}(h,k)), \end{aligned} \tag{6}$$

$$\begin{aligned} ((f^g)^h)^k &= \text{pow}(\text{pow}(\text{pow}(f,g),h),k) = \text{pow}(f, ghk), \\ &\neq \text{pow}(\text{pow}(f,g), \text{pow}(h,k)) = (f^g)^{(h^k)}, \\ &\neq \text{pow}(f, \text{pow}(g, \text{pow}(h,k))) = f^{(g^{(h^k)})}. \end{aligned} \tag{7}$$

2.2 Expansion of Identity

This section introduces a 'trick' we refer to as 'expansion of identity' (EOI) which allows expansion of tree representations in a way that can be helpful in computing jump expansions. In EOI we identify a bijection $\mathfrak{F} : \mathfrak{D} \to \mathfrak{C}$ that is a unary operation ($\dim \mathfrak{D} = \dim \mathfrak{C} = 1$). Any expression $f \in \mathfrak{D}$ can be represented as

$$f = \mathfrak{F}(\mathfrak{F}^{-1}(f)) = \mathfrak{F}^{-1}(\mathfrak{F}(f)). \tag{8}$$

Any n such bijections could be applied in sequence to achieve arbitrary expansion of f. The purpose of expanding f in this way is that certain pairs of bijection and function may yield a simplified $\mathfrak{F}(f)$ or $\mathfrak{F}^{-1}(f)$ that admits a convenient jump expansion. Consider, for instance, arbitrary exponentiation (binary operation), and the natural logarithm. We expand g^h for expressions g and h such that $g^h > 0$ to yield an exponential of a product of h and $\ln(g)$:

$$g^h = \exp(\ln g^h) = \exp(h \ln g). \tag{9}$$

3 Averages

3.1 Direct Averages

Direct two-point averages are shown in Table 2 along with their limits as $f_\ell, f_r \to f$.

Table 2: Direct averages, symbols, expressions, and limits when the left and right states are identical.

Average	Symbol	Expression	Limit as $f_\ell, f_r \to f$
arithmetic	$\overline{f}$	$(f_\ell + f_r)/2$	f
harmonic	$\acute{f}$	$(f_\ell f_r)/(f_\ell + f_r)$	$f/2$
geometric	$\breve{f}$	$\sqrt{f_\ell f_r}$	$\lvert f \rvert$
quadratic	$\ddot{f}$	$\sqrt{(f_\ell^2 + f_r^2)/2}$	f

3.2 Indirect Averages

This section details the series approximations to important 'special' indirect averages, such as the logarithmic average and exponential average. In general a special average is

$$[[f]]_{\mathrm{h}} = \frac{\Delta f}{\Delta h(f)}, \tag{10}$$

which can be expanded with a Taylor series as

$$\frac{1}{[[f]]_{\mathrm{h}}} = \frac{\Delta h(f)}{\Delta f} = \sum_{i=1}^{\infty} \frac{1}{i!} (\Delta f)^{i-1} \frac{\mathrm{d}^i h}{\mathrm{d} f^i}. \tag{11}$$

In checking consistency of entropy conservative fluxes, we need to evaluate special averages in the limit of $\Delta f \to 0$. Directly from the result above we have

$$\lim_{\Delta f \to 0} \frac{1}{[[f]]_{\mathrm{h}}} = \frac{\mathrm{d}h}{\mathrm{d}f}. \tag{12}$$

3.3 Simplification of Averages of Powers Into Powers of the Average

In this section we show how to simplify averages of powers (*e.g.* $\overline{f^2}$) into arithmetic averages, $\overline{f}$, and the product $\widetilde{f} = -f_\ell f_r$, which shows up frequently in computing jump expansions. This is relevant in particular cases when polynomials are used and dramatic simplifications (and computational cost reduction) can be gained by expressing quantities such as $\overline{f^2}$ in terms of $\overline{f}$ and $\widetilde{f}$. Before showing a more general derivation, consider the average of the square, $\overline{f^2}$, and the square of the average, $\overline{f}^{\,2}$:

$$\begin{aligned} \overline{f^2} &= \frac{1}{2}(f_\ell^2 + f_r^2), \\ \overline{f}^{\,2} &= \frac{1}{4}(f_\ell + f_r)^2 = \frac{1}{4}(f_\ell^2 + f_r^2 + 2 f_\ell f_r). \end{aligned} \tag{13}$$

These simple expansions show that we can relate the square of the average and the average of the square as

$$2\overline{f}^{\,2} = \overline{f^2} + f_\ell f_r = \overline{f^2} - \widetilde{f} \quad \to \quad \overline{f^2} = 2\overline{f}^{\,2} + \widetilde{f}. \tag{14}$$

Similarly, consider the cube:

$$\begin{aligned} \overline{f^3} &= \frac{1}{2}(f_\ell^3 + f_r^3), \\ \overline{f}^{\,3} &= \frac{1}{8}(f_\ell + f_r)^3 = \frac{1}{8}(f_\ell^3 + f_r^3 + 3f_\ell^2 f_r + 3f_\ell f_r^2), \end{aligned} \tag{15}$$

which yield

$$4\overline{f}^{\,3} = \frac{1}{2}(2\overline{f^3} + 3f_\ell f_r(f_\ell + f_r)) = \overline{f^3} - 3\widetilde{f}\,\overline{f} \quad \rightarrow \quad \overline{f^3} = 4\overline{f}^{\,3} + 3\widetilde{f}\,\overline{f}\,. \tag{16}$$

The algebra to derive these results for $\overline{f^2}$ and $\overline{f^3}$ is tedious and grows substantially more so with increasing exponent. To derive a general form for $\overline{f^n}$ for $n \in \mathbb{Z}^+$ we utilize the binomial theorem, which states

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}, \tag{17}$$

which is relevant to arithmetic averages as

$$\overline{f}^{\,n} = \left(\frac{f_\ell + f_r}{2}\right)^n = \frac{(f_\ell + f_r)^n}{2^n} = \frac{1}{2^n}\sum_{k=0}^{n} \binom{n}{k} f_\ell^k f_r^{n-k}. \tag{18}$$

The first and last terms in the sum may be separated out,

$$2^n \overline{f}^{\,n} = \left[f_\ell^n + f_r^n + \sum_{k=1}^{n-1} \binom{n}{k} f_\ell^k f_r^{n-k} \right] = \left[2\overline{f^n} + \sum_{k=1}^{n-1} \binom{n}{k} f_\ell^k f_r^{n-k} \right]. \tag{19}$$

Next, identify that all terms remaining in the summation have $f_\ell f_r$ in them, allowing us to write

$$2^n \overline{f}^{\,n} = \left[2\overline{f^n} + f_\ell f_r \sum_{k=1}^{n-1} \binom{n}{k} f_\ell^{k-1} f_r^{n-k-1} \right] = \left[2\overline{f^n} - \widetilde{f} \sum_{k=1}^{n-1} \binom{n}{k} f_\ell^{k-1} f_r^{n-k-1} \right]. \tag{20}$$

Now we may pull the first 'trick' of separating out the first and last terms in the summation that involve only x_l or x_r:

$$2^n \overline{f}^{\,n} = \left[2\overline{f^n} - \widetilde{f} \left(\binom{n}{1} \left(f_\ell^{n-2} + f_r^{n-2} \right) + \sum_{k=2}^{n-2} \binom{n}{k} f_\ell^{k-1} f_r^{n-k-1} \right) \right], \tag{21}$$

which introduces another average:

$$2^n \overline{f}^{\,n} = \left[2\overline{f^n} - \widetilde{f} \left(2\binom{n}{1} \overline{f^{n-2}} + \sum_{k=2}^{n-2} \binom{n}{k} f_\ell^{k-1} f_r^{n-k-1} \right) \right]. \tag{22}$$

Continuing on with the pattern, the next form is

$$2^n \overline{f}^{\,n} = \left[2\overline{f^n} - \widetilde{f} \left(2\binom{n}{1} \overline{f^{n-2}} - \widetilde{f} \sum_{k=2}^{n-2} \binom{n}{k} f_\ell^{k-2} f_r^{n-k-2} \right) \right], \tag{23}$$

$$2^n \overline{f}^{\,n} = \left[2\overline{f^n} - \widetilde{f} \left(2\binom{n}{1} \overline{f^{n-2}} - \widetilde{f} \left[2\binom{n}{2} \overline{f^{n-4}} + \sum_{k=3}^{n-3} \binom{n}{k} f_\ell^{k-2} f_r^{n-k-2} \right] \right) \right], \tag{24}$$

which can be written as

$$2^{n-1}\overline{f}^{\,n} = \overline{f^n} - \binom{n}{1}\widetilde{f}\,\overline{f^{n-2}} + \binom{n}{2}\widetilde{f}^2\,\overline{f^{n-4}} - \binom{n}{3}\widetilde{f}^3\,\overline{f^{n-6}} + \dots . \tag{25}$$

We can write the general case as

$$\overline{f}^{\,n} = \frac{1}{2^{n-1}}\sum_{j=0}^{2j\le n}\binom{n}{j}\left(-\widetilde{f}\right)^j\overline{f^{n-2j}} = \frac{1}{2^{n-1}}\left(\overline{f^n} + \sum_{j=1}^{2j\le n}\binom{n}{j}\left(-\widetilde{f}\right)^j\overline{f^{n-2j}}\right), \tag{26}$$

which allows the following solution expression for $\overline{f^n}$,

$$\overline{f^n} = 2^{n-1}\overline{f}^{\,n} - \sum_{j=1}^{2j\le n}\binom{n}{j}\left(-\widetilde{f}\right)^j\overline{f^{n-2j}}. \tag{27}$$

This yields the following results for n up to 7:

$$\begin{aligned}
\overline{f} &= \overline{f}\,,\\
\overline{f^2} &= 2\left(\overline{f}^{\,2} + \widetilde{f}\right),\\
\overline{f^3} &= 4\overline{f}^{\,3} + 3\widetilde{f}\overline{f}\,,\\
\overline{f^4} &= 2\left(4\left[\overline{f}^{\,4} + \widetilde{f}\overline{f}^{\,2}\right] + \widetilde{f}^2\right),\\
\overline{f^5} &= 16\overline{f}^{\,5} + 20\widetilde{f}\overline{f}^{\,3} + 5\widetilde{f}^2\overline{f}\,,\\
\overline{f^6} &= 2\left(16\overline{f}^{\,6} + 24\widetilde{f}\overline{f}^{\,4} + 9\widetilde{f}^2\overline{f}^{\,2} + \widetilde{f}^3\right).\\
\overline{f^7} &= 64\overline{f}^{\,7} + 112\widetilde{f}\overline{f}^{\,5} + 56\widetilde{f}^2\overline{f}^{\,3} + 7\widetilde{f}^3\overline{f}\,.
\end{aligned} \tag{28}$$

The average of a polynomial can now be expressed similarly:

$$\mathfrak{p}(f) = \sum_{i=0}^{m} a_i f^i, \tag{29}$$

$$\overline{\mathfrak{p}(f)} = \sum_{i=0}^{m} a_i\overline{f^i} = \sum_{i=0}^{m} a_i 2^{i-1}\overline{f}^{\,i} - \sum_{i=0}^{m} a_i\left(\sum_{j=1}^{2j\le i}\binom{i}{j}\left(-\widetilde{f}\right)^j\overline{f^{i-2j}}\right), \tag{30}$$

although it isn't immediately clear how one might simplify this expression further. . .

4 Jump Expansions

This section details jump expansions of the fundamental elements of expression trees enumerated in §2.1. Chaining the elemental jump expansions via binary operators as discussed in §2 then allows automated jump expansion of arbitrary expressions.

The 'jump' of a function is simply the difference across the interface,

$$\Delta f = f_r - f_\ell, \tag{31}$$

where in a finite volume context f_r is outside of the a cell and f_ℓ is inside of the cell. The jump ratio of a function f with respect to auxiliary variable a is

$$\mathcal{R}_a^f = \frac{\Delta f}{\Delta a} = \frac{f_r - f_\ell}{a_r - a_\ell}. \tag{32}$$

Note that as $\Delta a \to 0$ we obtain the partial derivative:

$$\lim_{\Delta a \to 0} \mathcal{R}_a^f = \frac{\partial f}{\partial a}. \tag{33}$$

The jump expansion of a function $f(a_1, a_2, \ldots)$, denoted $\mathcal{E}\left(f\right)$, is the set of the pairs of variable and jump ratio,

$$\mathcal{E}\left(f\right) = \left\{ \left(a, \mathcal{R}_a^f\right) : a \in \mathcal{A} \right\}. \tag{34}$$

First, the jump expansion of a constant c is trivial:

$$\mathcal{R}_a^c = 0 \quad \forall a \in \mathcal{A}. \tag{35}$$

Next, the jump expansion of an auxiliary variable is simply a Kronecker delta,

$$\mathcal{R}_{a_j}^{a_i} = \delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases} \tag{36}$$

Binary operations between arbitrary expressions f and g have the following jump ratios:

$$\mathcal{R}_a^{f+g} = \frac{\Delta f + \Delta g}{\Delta a} = \frac{\Delta f}{\Delta a} + \frac{\Delta g}{\Delta a} = \mathcal{R}_a^f + \mathcal{R}_a^g, \tag{37}$$

$$\mathcal{R}_a^{f-g} = \mathcal{R}_a^f - \mathcal{R}_a^g, \tag{38}$$

$$\mathcal{R}_a^{fg} = \overline{f}\,\mathcal{R}_a^g + \overline{g}\,\mathcal{R}_a^f, \tag{39}$$

$$\mathcal{R}_a^{f/g} = \mathcal{R}_a^{f\,\mathrm{rec}g} = \overline{f}\,\mathcal{R}_a^{\mathrm{rec}g} + \overline{\mathrm{rec}g}\,\mathcal{R}_a^f, \tag{40}$$

$$\mathcal{R}_a^{f^g} = \frac{1}{[[g \ln f]]_{\exp}} \left(\overline{\ln f}\,\mathcal{R}_a^g + \frac{\overline{g}}{\widehat{f}} \mathcal{R}_a^f \right), \quad f^g > 0. \tag{41}$$

Multiplication of more than two operands is treated as in (6), which one might refer to as 'left-to-right.'[2]

$$\begin{aligned} \mathcal{R}_a^{fghk} &= \overline{ghk}\,\mathcal{R}_a^f + \overline{f}\,\left(\overline{hk}\,\mathcal{R}_a^g + \overline{g}\,\left[\overline{k}\,\mathcal{R}_a^h + \overline{h}\,\mathcal{R}_a^k \right] \right), \\ &= \overline{ghk}\,\mathcal{R}_a^f + \overline{f}\,\overline{hk}\,\mathcal{R}_a^g + \overline{f}\,\overline{g}\,\overline{k}\,\mathcal{R}_a^h + \overline{f}\,\overline{g}\,\overline{h}\,\mathcal{R}_a^k, \end{aligned} \tag{42}$$

The n-ary case for exponentiation is shown below.

$$\mathcal{R}_a^{((f^g)^h)^k} = \mathcal{R}_a^{f^{ghk}} = \frac{1}{[[ghk \ln f]]_{\exp}} \left(\overline{\ln f}\,\mathcal{R}_a^{ghk} + \frac{\overline{ghk}}{\widehat{f}} \mathcal{R}_a^f \right). \tag{43}$$

Unary operations are considered generally with indirect averages except for the negative, reciprocal, and exponentiation operations. We define the 'reciprocal average' $\widetilde{f}$ here.

$$\mathcal{R}_a^{\mathrm{uop}(f)} = \frac{\mathcal{R}_a^f}{[[f]]_{\mathrm{uop}}}, \tag{44}$$

$$\mathcal{R}_a^{\mathrm{neg}(f)} = -\mathcal{R}_a^f, \tag{45}$$

$$\mathcal{R}_a^{\mathrm{rec}(f)} = \frac{\mathcal{R}_a^f}{[[f]]_{\mathrm{rec}}} = -\frac{\mathcal{R}_a^f}{2\overline{f}\,\acute{f}} = -\frac{\mathcal{R}_a^f}{f_\ell f_r} = \frac{\mathcal{R}_a^f}{\widetilde{f}}, \tag{46}$$

[2]One may group together quantities in products in `Relleno` by creating special expressions.

$$\mathcal{R}_a^{\log_r(f)} = \frac{1}{\ln(r)} \frac{\mathcal{R}_a^f}{\widehat{f}}, \quad r \in \mathbb{R}^+, \tag{47}$$

$$\mathcal{R}_a^{r^g} = \frac{\ln r}{[[g \ln r]]_{\text{exp}}} \mathcal{R}_a^g, \quad r \in \mathbb{R}^+. \tag{48}$$

Finally we consider the exponentiation of a function, for each case described in Table 1. For case A (arbitrary (non-integer) power $r_+ \in \mathbb{R}^+$), we have

$$\text{case A:} \quad \mathcal{R}_a^{f^{r_+}} = \frac{r_+ \mathcal{R}_a^f}{[[r_+ \ln f]]_{\text{exp}} \widehat{f}}. \tag{49}$$

Case B is treated as the reciprocal of case A.

$$\text{case B:} \quad \mathcal{R}_a^{f^{r_-}} = \mathcal{R}_a^{\text{rec}(f^{r_+})} = \frac{\mathcal{R}_a^{f^{r_+}}}{[[f^{r_+}]]_{\text{rec}}}. \tag{50}$$

Next, consider cases C, D, E, and F with integer powers. Starting off, we give the following simple result for the square:

$$\mathcal{R}_a^{f^2} = 2\overline{f}\,\mathcal{R}_a^f. \tag{51}$$

For the cube, one simply splits an expression into a product of the quantity and the square to arrive at[3]

$$\mathcal{R}_a^{f^3} = \left(\overline{f^2} + 2\overline{f}\right) \Delta f a. \tag{52}$$

For the quartic, f^4, and higher exponents, we are stuck with the fact that we could decompose f^{n_+} into $f f^{n_+ - 1}$ or any other combination of exponents. The most cost-efficient method is to split even powers into squares and odd powers into a product of the function and the lower even power. Thus, case C (even power) is handled as such:

$$\text{case C:} \quad \mathcal{R}_a^{f^{2k}} = \mathcal{R}_a^{(f^k)^2} = 2\overline{f^k}\,\mathcal{R}_a^{f^k} = 2\overline{f^k}\,\mathcal{R}_f^{f^k}\,\mathcal{R}_a^f. \tag{53}$$

And case E (odd power) is

$$\text{case E:} \quad \mathcal{R}_a^{f^{2k+1}} = \overline{f}\,\mathcal{R}_a^{(f^k)^2} + \overline{f^{(2k)}}\,\mathcal{R}_a^f = \left(2\overline{f}\,\overline{f^k}\,\mathcal{R}_f^{f^k} + \overline{f^{(2k)}}\right) \mathcal{R}_a^f. \tag{54}$$

For negative integer powers (case D and F), we use the following decomposition:

$$\mathcal{R}_a^{f^{n_-}} = \mathcal{R}_{(1/f)}^{(1/f)^{n_+}}\,\mathcal{R}_f^{1/f}\mathcal{R}_a^f = \frac{1}{\widetilde{f}}\mathcal{R}_{(1/f)}^{(1/f)^{n_+}}\,\mathcal{R}_a^f, \tag{55}$$

where $\mathcal{R}_{(1/f)}^{(1/f)^{n_+}}$ is handled as in case C or E.

Finally for cases G and H, we handle these as if the power is an arbitrary real number, except for the square root, for which we use the form below derived from the jump expansion of the square:

$$\mathcal{R}_a^{\sqrt{f}} = \frac{1}{2\overline{\sqrt{f}}}\mathcal{R}_a^f \quad (f > 0), \tag{56}$$

[3]Note that using the formulas in §3.3 yields $\mathcal{R}_a^{f^3} = 2\left(2\overline{f} + \widetilde{f}\right)\Delta f a$, which is 50% faster to compute, as it involves only two additions and multiplications while the form with $\overline{f^2}$ involves three of each.

Note that the following form exists for the cube root, although we do not use it. Forms for larger powers likely involve more and more complicated roots, making their efficient evaluation difficult, and are not pursued here.

$$\mathcal{R}_a^{\sqrt[3]{f}} = \frac{1}{2\left(\overline{\sqrt[3]{f}}\right)^2 + \overline{\left(\sqrt[3]{f}\right)^2}} \mathcal{R}_a^f. \tag{57}$$