diman: A Clojure Package for Dimensional Analysis

Lungsi Sharma[*1]

1 Ronin Institute

DOI: 10.21105/joss.03735

Software

- Review
- Repository
- Archive

Editor: Matthew Sottile

Reviewers:

- @oyvinht
- @khinsen

Submitted: 30 July 2021
Published: 14 January 2022

Summary

`diman` (**dim**ensional **an**alysis) is a Clojure based scientific software package with the ability to: create dimensional formulas, create dimensional equations, check dimensional homogeneity (consistency), and derive dimensionless products.

`diman` provides functions for each step of the analytic process for checking dimensional homogeneity or deriving dimensionless products; the repetitive operations (computational) are hidden. Users can write compound functions that perform a desired process. Thus, not only is the computational labor saved, but also introspection of the analysis is possible; the analyst is able to go through the steps of dimensional analysis.

Statement of need

Explaining the mechanism of a phenomenon is often the goal of experiments. As most mechanistic description is expressible in terms of some measurable quantity, its value is a function of other measurable quantities; the function represents the relationship among the quantities, which provides a mechanistic explanation. For example, $F = ma = m\frac{dv}{dt}$ where the measurable value of force F is a function of the measurable quantities: mass, m; velocity, v; and time, t.

Some or all the independent variables of the parent (first or original) functions have dimensions. Since most of the functions are unknown, and hence conceptual, the researcher deals with many candidates for independent variable, whose considerations are based on experimental results. Although the mathematical expression of the function is unknown, knowledge of the relationship among the measurable quantities is profitable not only in putting together the series of experimental results to explain the mechanism, but also testing the hypothesis presented by the function.

If possible, it is beneficial to use the transformed parent function, where all the independent variables are dimensionless. Dimensionless products are magnitudes that contains information on the dimensional quantities that they are a product of. Therefore, not only are points in a graph of dimensionless products experimentally determinable, but also they are more informative than dimensional graphs. Reducing the number of independent variables to a smaller collection of dimensionless products can assist in understanding the mechanism of the phenomenon (Langhaar, 1951; Sharma, 2021).

Numerous software packages have been developed to deal with dimensions in some shape or form (Preussner, 2018; Sharma, 2021). Most incorporate the ability to tag quantities with units, however, few are capable of doing consistency checks and fewer still deal with dimensionless products let alone, deriving dimensionless products.

*co-first author

Frink and F sharp are two active languages that incorporate units of measurement. Frink is a calculating tool and programming language designed to make physical calculations as it tracks units of measure throughout calculations (Eliasen, 2004). F# is a general purpose language that allows annotating floating point and integer values with statically-typed unit metadata. F#'s units of measure is based on a prototype Meta Language (ML) like language that has dimensional type; the language supports type polymorphism as well as dimension polymorphism (Kennedy, 1996).

In F# the notion of dimension is based on ideas from type theory; therefore, dimensions satisfy algebraic properties of Abelian groups (commutative groups). The concepts of dimensional type and dimensional invariance can be mathematically extended to apply Buckingham's Theorem (also called Pi theorem) (Kennedy, 1996)—this theorem is the basis for deriving the set of dimensionless products (Ngwua, 2020).

`diman` is designed with an emphasis on **analysis**; the application of the algebraic theory of dimensionally homogeneous functions (Langhaar, 1951). Unlike F#'s units of measure which performs dimensional checks during calculations, checks for dimensional homogeneity in diman is manual. Also, the possibility of applying Buckingham's theorem in F# is done by mathematical abstraction (Kennedy, 2010) while diman provides functions to derive the complete set of dimensionless products of a given equation. Therefore, diman is easily put into practice for providing complete information of experimental results of physical systems in a compact form and also transform hypothetical function that describes the physical system (Sharma, 2021).

Design and implementation

Based on the International System of Units, `diman` uses the seven base (or elementary) dimensions: [M], [L], [T], [A], [K], [mol] and [cd] for the quantities: mass, length, time, electric current, thermodynamic temperature, amount of substance and luminous intensity respectively (BIPM, 2020). They are defined in `base_dimensions`. Furthermore, some well-known dimensions derived from the `base_dimensions` are defined in `standard_formula`; a dimensional formula for respective quantity is its dimension.

Consistency checking

This is done by the predicate `consistent?`. There are some preliminary steps before invoking the predicate. Consider the given function $E = \frac{1}{2}mv^2$.

We define the variables

```
=> (def variables [{:symbol "E", :quantity "energy"}
                   {:symbol "m", :quantity "mass"}
                   {:symbol "v", :quantity "velocity"}])
```

then the equation

```
=> (def equation {:lhs "E^(1)", :rhs "0.5*m^(1)*v^(2)"})
```

Finally, the predicate `consistent?` is used to check if the equation is dimensionally homogenous.

```
=> (consistent? variables equation)
true
```

Derivation of set of dimensionless products

Imagine that the study of a system results in a hypothesis such that some measurable dimensionless product is a homogeneous function f of the independent variables P, Q, R, S, T, U and V. Also, assume that the independent variables have dimensions such that

```
=> (def dimensional_formulae_of_all_independent_variables
        [{:quantity "term-p", :dimension "[M^(2)*L^(1)]"}
         {:quantity "term-q", :dimension "[M^(-1)*T^(1)]"}
         {:quantity "term-r", :dimension "[M^(3)*L^(-1)]"}
         {:quantity "term-s", :dimension "[T^(3)]"}
         {:quantity "term-t", :dimension "[L^(2)*T^(1)]"}
         {:quantity "term-u", :dimension "[M^(-2)*L^(1)*T^(-1)]"}
         {:quantity "term-v", :dimension "[M^(1)*L^(2)*T^(2)]"}])
```

Supposing the independent variables of the parent function f are not already defined in `standard_formula`, inject the dimensions of the independent variables into the `standard_formula` for the present read–eval–print loop session by

```
=> (update-sformula dimensional_formulae_of_all_independent_variables)
```

Thus, `diman` now contains dimensions of the independent variables of f. Hence, the independent variables can be defined as

```
=> (def independent_variables
        [{:symbol "P", :quantity "term-p"}
         {:symbol "Q", :quantity "term-q"}
         {:symbol "R", :quantity "term-r"}
         {:symbol "S", :quantity "term-s"}
         {:symbol "T", :quantity "term-t"}
         {:symbol "U", :quantity "term-u"}
         {:symbol "V", :quantity "term-v"}])
```

The theory of dimensionless products (Ngwua, 2020) tells us that the derivation of dimensionless products can be broken down into four steps: generate the dimensional matrix, solve the homogeneous equation, determine the solution matrix and get the set of dimensionless products. Compounding the first three steps into one code block we get,

```
=> (def solution_matrix
        (get-solution-matrix
            (solve (get-augmented-matrix
                        (generate-dimmat independent_variables)))))
```

This is the solution matrix for a complete set of dimensionless products.

```
=> (view-matrix solution_matrix)
[1 0 0 0 -11N 5N 8N]
[0 1 0 0 9N -4N -7N]
[0 0 1 0 -9N 5N 7N]
[0 0 0 1 15N -6N -12N]
Size -> 4 x 7
```

The set of dimensionless products can be obtained from the solution matrix by using the function `get-dimensionless-products`. Thus

```
=> (println (get-dimensionless-products solution_matrix independent_variables))
  [{:symbol "pi0", :expression "P^(1)*T^(-11)*U^(5)*V^(8)"}
  {:symbol "pi1", :expression "Q^(1)*T^(9)*U^(-4)*V^(-7)"}
  {:symbol "pi2", :expression "R^(1)*T^(-9)*U^(5)*V^(7)"}
  {:symbol "pi3", :expression "S^(1)*T^(15)*U^(-6)*V^(-12)"}]
```

or

$$\pi_0 = PT^{-11}U^5V^8, \quad \pi_1 = QT^9U^{-4}V^{-7}, \quad \pi_2 = RT^{-9}U^5V^7, \quad \pi_3 = ST^{15}U^{-6}V^{-12}$$

Therefore, function f is transformed into some function f_1 whose independent variables are the dimensionless products; π_0, π_1, π_2, and π_3—π is the conventional notation for any dimensionless product. Thus, the number of variables is reduced from 7 to 4.

Conclusion

`diman` is a Clojure library with no other dependencies. It has its own linear algebra submodule that provides all the necessary operations. Internally, the numerical data type is Clojure's *ratio*; a ratio between integers rather than floats (Hickey, 2021). This avoids truncation and rounding errors. Since dimensional analysis does not often involve very large matrices, the hit on computational performance due to using the *ratio* number type is practically insignificant. `diman` supplies all the necessary functions for dimensional homogeneity operations and the derivation of dimensionless products; thus making the analysis steps transparent.

Acknowledgements

The project received no funding.

References

BIPM. (2020). *Base unit definitions*. Bureau International des Poids et Mesures. https://www.bipm.org/en/measurement-units/base-units.html

Eliasen, A. (2004). *Frink - A Language for Understanding the Physical World*. https://frinklang.org/LL4.html

Hickey, R. (2021). *Clojure - Data Structures*. Clojure. https://clojure.org/reference/data_structures

Kennedy, A. J. (1996). *Programming languages and dimensions* (No. 391). University of Cambridge. https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-391.pdf

Kennedy, A. J. (2010). Types for Units-of-Measure: Theory and Practice. In Z. Horváth, R. Plasmeijer, & V. Zsók (Eds.), *Central European Functional Programming School: Third Summer School, CEFP 2009, Budapest, Hungary, May 21-23, 2009 and Komárno, Slovakia, May 25-30, 2009, Revised Selected Lectures* (pp. 268–305). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-17685-2_8

Langhaar, H. L. (1951). *Dimensional Analysis and Theory of Models*. John Wiley & Sons, Inc.

Ngwua, L. (2020). Theory of Dimensionless Products. In *Neuralgraphs*. https://www.neuralgraphs.com/lectures/diman/lectp1.html

Preussner, G. M. (2018). Dimensional Analysis in Programming Languages. In *Personal Homepage*. https://gmpreussner.com/research/dimensional-analysis-in-programming-languages

Sharma, B. L. (2021). *Dimensional analysis: A potential use of dimensionless products in biology/neuroscience*. Institut des Neurosciences Paris-Saclay (NeuroPSI), Département de Neurosciences Intégratives et Computationnelles (ICN), CNRS. https://lungsi-slides.github.io/dataclub/icn/2021July26/index.html