Software Requirements Specification for Drasil Matrix, Vector and Tensor Extension

Christopher W. Schankula

January 24, 2025

Contents

Revision History

Date	Version	Notes
January 23rd	1.0	Initial Work on Document for Presentation
Date 2	1.1	Notes

1 Reference Material

This section records information for easy reference.

1.1 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	unit	description
A_C	m^2	coil surface area
A_{in}	m^2	surface area over which heat is transferred in

[Use your problems actual symbols. The si package is a good idea to use for units. —TPLT]

1.2 Terminology

- **Specification-time**: A property that holds true when a specification is created in the Drasil framework.
- **Generation-time**: A property that holds true or is checked when the code/documentation is generated by the Drasil system.
- **Runtime**: A property that holds true or is checked when the generated code is run.

1.3 Abbreviations and Acronyms

symbol	description
A	Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
Drasil Matrix, Vector and Tensor Extension	[put an expanded version of your program name here (as a
TM	Theoretical Model

[Add any other abbreviations or acronyms that you add —TPLT]

1.4 Mathematical Notation

2 Introduction

This introduction section states the purpose of this document, the scope of the requirements, the characteristics of the intended reader, an overview of the Drasil project, and describes the organization of the rest of the document.

2.1 Purpose of Document

The purpose of this document is to document the necessary mathematical background for, and the software requirements of, an extension of the Drasil project encoding tensor, vector, and matrix operations. It is intended to allow the different stakeholders to communicate about and iterate on the project in a formal way. The document will likely be updated throughout the project, and the changes will be recorded in the Revision History table.

2.2 Scope of Requirements

The scope of the requirements will be related to the addition of tensors, vectors, and matrices to the Drasil project.

2.3 Characteristics of Intended Reader

TODO

2.4 Overview of Drasil Project

Drasil is "a framework a framework for generating all of the software artifacts from a stable knowledge base, focusing currently on scientific software". The framework is written in Haskell and allows generation of Software Requirements Specifications, Python, Java, C-Sharp, and C++ code, README files, and Makefiles.

2.5 Organization of Document

The rest of the document is organized as follows: Section 3 presents related work in the area of tensors, Section 4 presents the mathematical definitions, transformation rules and allowed operations on tensors, vectors, and matrices. Then, Section 5 describes how vectors and matrices can be defined as special cases of tensors. Finally, Section 6 presents general requirements for the system extension as well as a set of "test-driven" requirements, denoting some example scientific problems to be encoded and solved using this new system, which will provide oracles with which to test its correctness.

3 Related Work

3.1 Tensors in Haskell

TODO: Tensor Hackage library

3.2 Tensors in Other Languages

TODO: Numpy, Tensorflow, etc.

4 Mathematical Definitions

This section provides mathematical definitions of tensors, vectors, and matrices. We begin with a definition of Einstein summation notation which will be used throughout the document to eliminate the need for explicit summations. Then we will turn our attention to vectors, matrices and generalized tensors. This includes terminology needed for each, the notation used to describe each one, the transformation rules governing each one, and the allowed operations we are targeting for the software. Note that vectors and matrices will be defined here without relying on their definition as a tensor; Section 5 will redefine them using tensors.

4.1 Einstein Summation Notation

Einstein summation notation is used to describe repeated summations and multiplications in a compact notation. They behave according to the following four rules (Faculty of Khan, 2023):

Rule 1: Any twice-repeated index in a single term is summed over.
For example, writing the term $a_{i,j}b_i$ represents the sum $\sum_{i=1}^{n} a_{i,j}b_i$. With Einstein notation, we can omit the sum notation.

Rule 2: A twice-repeated index is called a dummy index; a once-repeated index is called a free index.
In the example above, i is a dummy index — it can be renamed however you would like and it still means $1 \dots n$. However, j is a free index and has restrictions on naming.

Rule 3: No index may occur 3 or more times in a given term.
For example, the term $a_{i,i}b_i$ is not legal.

Rule 4: In an equation involving Einstein notation, the free indices on both sides must match.
Some examples of correctly-formed equations:

- $x_i = a_{i,j}b_i$ is valid because i is free on both the LHS and RHS
- $a_i = A_{k,i}B_{k,j}x_j + C_{i,k}u_k$ is valid because i is a free variable on the LHS, and in every term it is the free variable on the RHS.

Some examples of incorrectly-formed equations:

- $x_i = A_{j,i}$ is invalid because i is the only free variable on the LHS, but i and j are both free on the RHS.
- $x_j = Ai, ku_k$ is invalid because j is free on the LHS, but i is free on the RHS.
- $x_i = A_{i,k}u_k + c_j$ is invalid because i is free on the LHS, but on the RHS, one term has i free while the other term has j free.

The remainder of this document will use Einstein notation to enhance brevity and clarity.

4.2 Vectors

A vector is a quantity having a magnitude and a direction Wikipedia contributors (2024b). They are used in mathematics and physics problems, especially those involving

4.2.1 Terms

- **Component**: An individual element of a vector.
- **Magnitude**: The size or "length" of a vector.
- **Direction**: The way a vector points in space.
- **Basis Vector**:

4.2.2 Notation

We will write vectors using the boldface, such as $\mathbf{V}$. An example of a vector with three elements is $\mathbf{a} = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}$. This is called a *row vector*. An example of a column vector is $\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$.

4.2.3 Allowed Operations

This subsection defines the allowed operations on vectors.

Equality

Two vectors are said to be equal if they have the same magnitude and direction (Wikipedia contributors, 2025b). This can also be defined by equating their components, i.e.

$$\mathbf{a}_i = \mathbf{b}_i \ \forall \ i = 1 \dots n$$

Vector Addition

Two vectors with the same size can be added together. Adding them adds their components together. That is,

$$\mathbf{a} + \mathbf{b} = \mathbf{a}_i + \mathbf{b}_i$$

Scalar Multiplication The scalar multiplication of a vector is equivalent to multiplying the magnitude by that number. Equivalently, it is the same as multiplying each component by the number. That is, a vector $\mathbf{a}$ scaled by the real number r is denoted $r\mathbf{a}$ and is defined as:

$$(r\mathbf{a})_i = r\mathbf{a}_i, 1 \leq i \leq n$$

Dot Product

The *dot product* of two vectors each of size n is represented by

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{a}_i\mathbf{b}_i$$

.

Cross Product

The *cross product* $\mathbf{c}$ of 3-dimensional vectors $\mathbf{a}$ and $\mathbf{b}$ is denoted as $\mathbf{c} = \mathbf{a} \times \mathbf{b}$, and is defined by Einstein notation as (McGinty, 2012):

$$c_i = \epsilon_{i,j,k}\mathbf{a}_j\mathbf{b}_k, \text{ where } \epsilon_{1,2,3} = \epsilon_{2,3,1} = \epsilon_{3,1,2} = 1 \text{ and } \epsilon_{3,2,1} = \epsilon_{2,1,3} = \epsilon_{1,3,2} = -1$$

4.3 Matrices

A *matrix* is a rectangular table of numbers, symbols, or expressions with entries arranged in rows and columns (Wikipedia contributors, 2025c). For example, the following is a 2-by-3 matrix:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \end{bmatrix}$$

4.3.1 Terms

- **Size**: The number of rows and columns in a matrix. A matrix with m rows and n columns is called an $m \times n$ matrix, or an m-by-n matrix.
- **Dimensions**: In the above example, m and n are called the matrix's *dimensions*.
- **Entries**: An individual element in the matrix.

4.3.2 Notation

By convention, we refer to matrices using an uppercase boldface name and their components using the lowercase equivalent without using boldface. The symbol $\mathbf{a}_{i,j}$ represents the *entry* at the ith row and jth column of the matrix.

4.3.3 Allowed Operations

The following is the list of allowed operations on matrices:

Equality
Like vectors, two matrices $\mathbf{A}$ and $\mathbf{B}$ of size $m \times n$ are equal if and only if their entries are equal:

$$\mathbf{A}_{i,j} = \mathbf{B}_{i,j},\ \forall\ i = 1 \ldots m, j = 1 \ldots n$$

Addition
The addition of two $m \times n$ matrices is calculated as:

$$(\mathbf{A} + \mathbf{B})_{i,j} = \mathbf{A}_{i,j} + \mathbf{B}_{i,j}, 1 \leq i \leq m, 1 \leq j \leq n$$

Scalar Multiplication The scalar multiplication of a real number c and a matrix $\mathbf{A}$ is denoted as $c\mathbf{A}$ and is computed as:

$$(c\mathbf{A})_{i,j} = c\mathbf{A}_{i,j}, 1 \leq i \leq m, 1 \leq j \leq n$$

Subtraction
The subtraction of two $m \times n$ matrices is denoted as $\mathbf{A} - \mathbf{B}$, and is the addition of the scalar multiplication by -1:

$$(\mathbf{A} - \mathbf{B})_{i,j} = \mathbf{A}_{i,j} + (-1)\mathbf{B}_{i,j}, 1 \leq i \leq m, 1 \leq j \leq n$$

Transposition
The *transpose* of an $m \times n$ matrix is the $n \times m$ matrix $\mathbf{A}^T$ formed by turning rows into columns and vice versa:

$$(\mathbf{A}^T)_{i,j} = \mathbf{A}_{j,i}$$

Matrix Multiplication
The *matrix multiplication* of matrices $\mathbf{A}$ and $\mathbf{B}$, is defined when matrix $\mathbf{A}$ is of size $m \times p$ and matrix $\mathbf{B}$ is of size $p \times n$. Then, the resulting matrix $\mathbf{AB}$ is a matrix of size $m \times n$ such that

$$(\mathbf{AB})_{i,j} = a_{i,r}b_{r,j}$$

Row Operations TODO

1. Row addition
2. Row multiplication
3. Row switching

Submatrix TODO
A *submatrix* is obtained by deleting any collection or rows and/or columns.

4.4 Tensors

A *tensor* is an "algebraic object that describes a multilinear relationship between sets of algebraic objects such as vectors, scalars and even other tensors" (Wikipedia contributors, 2025d). While tensors are considered to be independent of any basis, they are often referred to by their components in a particular basis; in this case they are stored as a multidimensional array. We will take advantage of this definition since it is a straightforward way to define and store tensors.

4.4.1 Terms

This subsection denotes common terminology used to describe tensors:

- **Index**: One dimension of the tensor.
- **Components**: The numbers in the multidimensional array representing a tensor (see below).
- **Order**: The total number of indices needed to uniquely identify each of the components of a tensor.
- **Basis vector**: A set B of vectors in a vector space V such that every vector in V can be written as a unique linear combination of the vectors in B (Wikipedia contributors, 2024a)
- **Coordinate Transformation**:

4.4.2 Multidimensional Array Representation

A common and convenient representation of a tensor for a given basis is using a multidimensional array to store the tensor. These can be thought of as higher-dimensional matrices. For instance, the tensor $T_{i,j}$ is the tensor of order two; it has two indices i and j. This is also called a second-order tensor. If the indices i and j run from 1 to 3 (as they would for a three-dimensional problem), then this tensor could be represented as a two-dimensional array with 9 elements, e.g.:

$$\begin{bmatrix} \mathbf{T}_{1,1} & \mathbf{T}_{1,2} & \mathbf{T}_{1,3} \\ \mathbf{T}_{2,1} & \mathbf{T}_{2,2} & \mathbf{T}_{2,3} \\ \mathbf{T}_{3,1} & \mathbf{T}_{3,2} & \mathbf{T}_{3,3} \end{bmatrix}$$

Basis Vectors

Though tensors are supposed to be independent of their basis, this multidimensional array description has the bases "built-in", so a definition of a tensor this way is incomplete without including the basis vectors, which are usually denoted as $\mathbf{e}_j$.

Formal Definition TODO: define multidimensional array representation formally.

4.4.3 Transformation Rules

4.4.4 Allowed Operations

This subsection defines the allowed operations on tensors, including operations which do not change the shape of the tensor like addition, subtraction and scalar multiplication, and those that do like tensor product, contraction, and raising and lowering an index (Wikipedia contributors, 2025d).

Tensor Addition

TODO: two tensors of the same shape can be added together.

Scalar Multiplication

TODO: Like vectors and matrices, scalar multiplication multiplies the components of the tensor

Tensor Subtraction

TODO: Combination of addition and scalar multiplication by -1.

Tensor Product

TODO: Special tensor product, generalization of matrix multiplication.

Contraction

TODO: Define contraction

Raising or lowering an index

TODO: Define raising/lowering an index

5 Vectors and Matrices Defined as Tensors

5.1 Vectors

A vector is a first-order tensor. It is a tensor of rank 1.

5.1.1 Tensor Definition of Vectors

5.1.2 Allowed Operations as Tensors

5.2 Matrices

A matrix is a second-order tensor. It is a tensor of rank 2.

5.2.1 Tensor Definition of Matrices

5.2.2 Allowed Operations as Matrices

6 Requirements

This section contains requirements for the system, categorized into *general requirements* which are requirements the system must follow, and *test-driven requirements* which are example scientific problems the system must be able to generate proper documentation and code to solve.

6.1 General Requirements

R1. The system shall contain an internal representation of tensors.

R2. The system shall contain a smart constructor for vectors, represented internally as rank-1 tensors.

R3. The system shall contain a smart constructor for matrices, represented internally as rank-2 tensors.

R4. The system shall allow the specification of tensor, vector, and matrix operations with specification-time fixed and variable sizes.

R5. The system shall support at least the vector, matrix, and tensor operations defined in Section 4 of this document.

R6. The system shall allow the generation of documentation using Einstein notation for vector, matrix, and tensors.

R7. The system shall allow the generation of code to perform the operations for vectors, matrices, and tensors.

R8. The system shall ensure or check the validity of operations at specification time, generation-time, or runtime, as appropriate.

R9. The addition of these new features to Drasil shall not break any existing examples.

R10. The system shall support the generation of artifacts to solve the scientific computing problems listed in Section 6.2 of this document.

6.2 Test-Driven Requirements

This section describes several example uses of tensors, vectors, and matrices which will be used to test the system. Since these problems have oracles, both in the form of the required generated documentation as well as computationally in terms of the results of the generated code, they are considered to be "test-driven" requirements of the system. That is, the system must be able to generate proper artifacts that are satisfactory to these examples.

6.2.1 Linear Operations

TODO: Basic transformations using transformation matrices (scaling, rotation, etc.)

6.2.2 Slope Stability Problems

TODO: Slope stability problems are...

6.2.3 Stress-Strain Problems

Stress-strain problems are problems that quantify the stress on a given object, for example a metal beam. The resulting tensor is a second-order tensor with components $\sigma_{i,j}$ that relates unit-length direction vectors $\mathbf{e}$ to the *traction vectors* $\mathbf{T}^{(\mathbf{e})}$ across an imaginary surface perpendicular to $\mathbf{e}$(Wikipedia contributors, 2025a):

$$\mathbf{T}^{(\mathbf{e})} = \mathbf{e} \cdot \sigma \text{ or } \mathbf{T}_j^{(\mathbf{e})} = \sigma_{i,j} e_i$$

Thus, we can encode the tensor as a two-dimensional array:

$$\begin{bmatrix} \sigma_{1,1} & \sigma_{1,2} & \sigma_{1,3} \\ \sigma_{2,1} & \sigma_{2,2} & \sigma_{2,3} \\ \sigma_{3,1} & \sigma_{3,2} & \sigma_{3,3} \end{bmatrix}$$

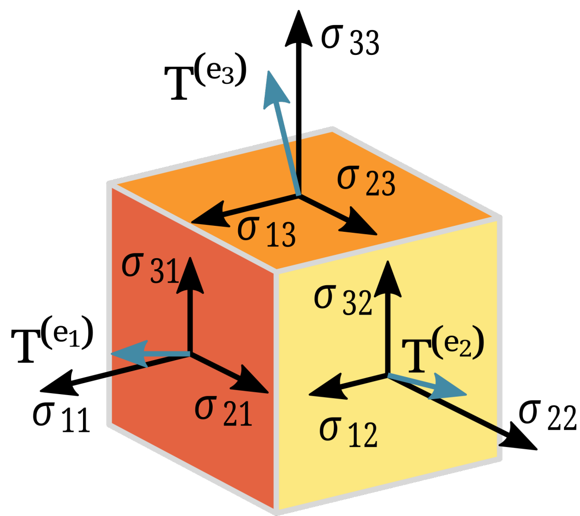

Figure 1: A graphical representation of a Cauchy stress tensor (Wikipedia contributors, 2025d). For any unit vector $\mathbf{v}$, the product $\mathbf{T} \cdot \mathbf{v}$ is a vector, denoted as $\mathbf{T}(\mathbf{v})$ that gives the force per unit area along the plane perpendicular to $\mathbf{v}$. This can be represented by a rank-2 tensor, with a total of 9 components.

References

Faculty of Khan. Clrtj7d2flm. https://www.youtube.com/watch?v=CLrTj7D2fLM, 2023. Accessed: 2025-01-23.

Bob McGinty. Tensor notation basics. https://www.continuummechanics.org/tensornotationbasic.html, 2012. Accessed: 2025-01-23.

Wikipedia contributors. Basis (linear algebra), October 2024a. URL https://en.wikipedia.org/wiki/Basis_(linear_algebra).

Wikipedia contributors. Vector (mathematics and physics), October 2024b. URL https://en.wikipedia.org/wiki/Vector_(mathematics_and_physics).

Wikipedia contributors. Cauchy stress tensor — wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Cauchy_stress_tensor, 2025a. Accessed: 2025-01-23.

Wikipedia contributors. Euclidean vector — wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Euclidean_vector, 2025b. Accessed: 2025-01-23.

Wikipedia contributors. Matrix (mathematics), January 2025c. URL https://en.wikipedia.org/wiki/Matrix_(mathematics).

Wikipedia contributors. Tensor, January 2025d. URL https://en.wikipedia.org/wiki/Tensor.