Aerospace Simulation

Tamas Kis │ tamas.a.kis@outlook.com

Tamas Kis
https://tamaskis.github.io

Contents

I   Kinematics

II Dynamics

III Orbital Mechanics

IV   Appendices

List of Algorithms

Angles

Rotations

Kinematics

Orbital Mechanics

Time Units

Time Scales

Preface

THE primary goal of this text is to bridge the gap between theory and implementation for aerospace simulations. The end result of all discussions is an algorithm (or a set of algorithms) that presents a clear computation procedure that can be implemented in any programming language. A lot of theory behind the algorithms *is* covered, but I choose to focus more on developing procedures that can be easily implemented, and less on rigorously explaining every little mathematical detail.

Notation

Coordinate Frames

$\mathcal{A}, \mathcal{B}, \mathcal{C}$	generic coordinate frames
$\mathcal{I}$	inertial frame
$\mathcal{R}$	rotating frame
$\mathcal{W}$	world frame
$\mathcal{B}$, body	body frame
eci	Earth-centered inertial frame
ecef	Earth-centered Earth-fixed frame
rtn	RTN (radial, tangential, normal) frame

Rotations

$\mathbf{R}_1$	rotation matrix for passive rotation about 1st axis
$\mathbf{R}_2$	rotation matrix for passive rotation about 2nd axis
$\mathbf{R}_3$	rotation matrix for passive rotation about 3rd axis
$\mathbf{R}_{321}$	rotation matrix for passive 3-2-1 rotation sequence
$\mathbf{R}_{313}$	rotation matrix for passive 3-1-3 rotation sequence
$\mathbf{R}_{\mathcal{A} \to \mathcal{B}}$	rotation matrix for passive rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$
$\mathbf{q}_{\mathcal{A} \to \mathcal{B}} = (q_0, q_1, q_2, q_3)^T$	quaternion for passive rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$ (scalar-first convention)
$\mathbf{e}_{\mathcal{A} \to \mathcal{B}}$	principal rotation vector for passive rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$
$\Phi_{\mathcal{A} \to \mathcal{B}}$	principal angle for passive rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$
$\psi_{\mathcal{A} \to \mathcal{B}}$	yaw angle for passive rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$ using 3-2-1 rotation sequence
$\theta_{\mathcal{A} \to \mathcal{B}}$	pitch angle for passive rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$ using 3-2-1 rotation sequence
$\phi_{\mathcal{A} \to \mathcal{B}}$	roll angle for passive rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$ using 3-2-1 rotation sequence

s	scalar
$\mathbf{s}$	column (mathematical) vector
$\boldsymbol{s}$	physical vector
$^{\mathrm{cf}}\boldsymbol{s}$	physical rate vector relative to the CF coordinate frame
$\dot{\mathbf{s}}$	time derivative of the vector $\mathbf{s}$ TODO
$\mathbf{s}_{\mathrm{cf}}$	coordinate vector of $\boldsymbol{s}$ expressed in the CF coordinate frame
$\mathbf{s}$	vector resolved in principal frame
$\hat{\mathbf{s}}$	unit vector of the column vector $\mathbf{a}$
$\hat{\boldsymbol{s}}$	unit vector of the physical vector $\mathbf{a}$
$\mathbf{A}$	matrix
$\boldsymbol{A}$	physical matrix
$\mathbf{I}_{n \times n}$	$n \times n$ identity matrix
$\boldsymbol{I}$	inertia tensor
$\mathbf{I}$	principal inertia tensor
$^{\mathrm{cf}}\mathbf{I}$	inertia tensor relative to the CF coordinate frame

PART I

Kinematics

1

Column Vectors

1.1 Column Vectors and Matrices

In mathematics, a **scalar** is a single number. An example of a scalar is

$$a = 5 \in \mathbb{R}$$

Convention 1: Notation for scalars.

Scalars are written using lowercase or uppercase, italic, non-boldface symbols. For example, a real-valued scalar may be written as

$$a \in \mathbb{R}$$

In mathematics, a **matrix** is a rectangular array of numbers. An example of a matrix is

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 6 & 0 & 5 \end{bmatrix} \in \mathbb{R}^{2 \times 3}$$

Convention 2: Notation for matrices.

Matrices are written using uppercase, upright, boldface symbols. For example, a real-valued matrix of size $m \times n$ may be written as

$$\mathbf{A} \in \mathbb{R}^{m \times n}$$

In mathematics, a **vector** is a list of numbers. An example of a vector is

$$\mathbf{v} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \in \mathbb{R}^4$$

In this text, we refer to such a vector as a **column vector**. Note that it is standard to assume that all vectors are **column**

vectors. If a vector is arranged in a row, we refer to it explicitly as a **row vector**, and denote its size differently. For example, an n-dimensional real-valued row vector is denote as being a member of the vector space $\mathbb{R}^{1 \times n}$. An example of a row vector is

$$\mathbf{v} = \begin{bmatrix} 1 & 2 & 3 & 4 \end{bmatrix} \in \mathbb{R}^{1 \times 4}$$

> **Convention 3:** Notation for column vectors.
>
> Column vectors are written using lowercase, upright, boldface symbols. For example, a real-valued vector of length n may be written as
>
> $$\mathbf{v} \in \mathbb{R}^n$$

Real-valued column vectors have both a magnitude and a direction. The **magnitude** of column vector $\mathbf{v}$ is calculated using the 2-norm.

$$\boxed{\|\mathbf{v}\| = v = \text{magnitude of } \mathbf{v}} \tag{1.1}$$

The **2-norm** of a column vector is defined as

$$\boxed{\|\mathbf{v}\| = \sqrt{\mathbf{v}^T \mathbf{v}}} \tag{1.2}$$

where the "T" denotes the transpose operation.

> **Convention 4:** Notation for the magnitude of a column vector.
>
> Since the magnitude of a column vector is a scalar, it is written using the italic, non-boldface version of the same symbol. For example, the magnitude of $\mathbf{v}$ is written as
>
> $$v$$

To describe the **direction** of a column vector, we use unit vectors. A **unit vector** is a vector of magnitude 1. Therefore, if we multiply a scalar quantity by a unit vector, we get a vector whose magnitude is equal to the scalar and whose direction is parallel to the unit vector's. The unit vector in the direction of $\mathbf{v}$ is defined as

$$\boxed{\hat{\mathbf{v}} = \frac{\mathbf{v}}{\|\mathbf{v}\|} = \frac{\mathbf{v}}{v} = \text{direction of } \mathbf{v}} \tag{1.3}$$

> **Convention 5:** Notation for mathematical unit vectors.
>
> A mathematical unit vector in the direction of a column vector is written using the same symbol but with a hat over it. For example, the unit vector in the direction of $\mathbf{v}$ is written as
>
> $$\hat{\mathbf{v}}$$

By rearranging Eq. (1.3), we can write a vector $\mathbf{v}$ in terms of its magnitude and direction.

$$\boxed{\mathbf{v} = v\hat{\mathbf{v}}} \tag{1.4}$$

1.2 **Coordinate Systems**

A **coordinate system** is a system that uses one or more numbers (i.e. coordinates) to determine the position of some geometric element with respect to that coordinate system. We use the following convention to name coordinate systems:

> **Convention 6:** Naming coordinate systems.
>
> Consider a coordinate system with origin O and axes $x_1, ..., x_n$. We refer to this coordinate system as $Ox_1...x_n$.

As a visual example, consider the three-dimensional coordinate system $Ox_1x_2x_3$ is illustrated in Fig. 1.1.
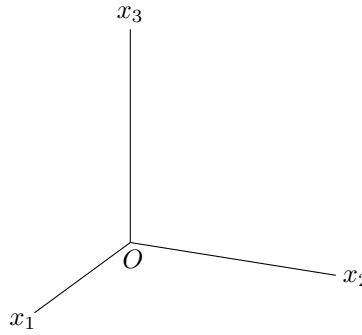
Figure 1.1: $Ox_1x_2x_3$ coordinate system.

A coordinate system is defined using an **ordered basis**, or simply a **basis** (since most bases we deal with in this text are ordered bases). Let X be the basis defining the $Ox_1...x_n$ coordinate system. Then

$$\boxed{X = (\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_n) = \text{ordered basis}} \tag{1.5}$$

> **Convention 7:** Ordered basis notation.
>
> An ordered basis is denoted using an uppercase, italic character, for example
>
> $$X$$

$\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_n$ are referred to as **basis vectors** since they define an ordered basis. In this text, we assume that all basis vectors are unit vectors, i.e. they all a of have magnitude 1[1].

$$\boxed{\|\hat{\mathbf{x}}_1\| = \cdots = \|\hat{\mathbf{x}}_n\| = 1} \tag{1.6}$$

Also note that all basis vectors have dimension equal to the dimension of the coordinate system they collectively define. For an n-dimensional coordinate system [47][51, pp. 157–162],

$$\boxed{\hat{\mathbf{x}}_i \in \mathbb{R}^n \quad \forall\, i = 1, ..., n} \tag{1.7}$$

---

[1] This is not required of basis vectors, but in the context of this text we make this assumption since most bases we will deal with have unit basis vectors

In our three-dimensional example, the tails of $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$ are all located at the origin O, while their heads points in the directions of the x_1, x_2, and x_3 axes, respectively. The $Ox_1x_2x_3$ coordinate system, together with its basis vectors $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$, is depicted in Fig. 1.2.
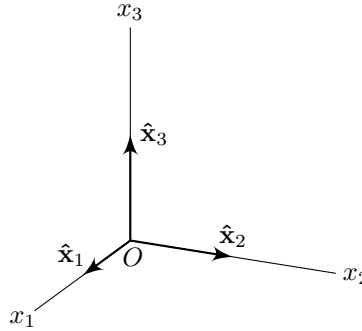
Figure 1.2: $Ox_1x_2x_3$ coordinate system with basis vectors.

1.3   Coordinate Vectors

Consider an arbitrary, n-dimensional column vector, $\mathbf{v} \in \mathbb{R}^n$. Let $X = (\mathbf{x}_1, ..., \mathbf{x}_n)$ be a basis defining the $Ox_1...x_n$ coordinate system. Then $\mathbf{v}$ can be written in terms of the basis vectors of the X coordinate system as

$$\mathbf{v} = \sum_{i=1}^{n} v_{x_i} \hat{\mathbf{x}}_i \tag{1.8}$$

where v_{x_i} is the **component** or **coordinate** of $\mathbf{v}$ with respect to the x_i axis (i.e. in the direction defined by $\hat{\mathbf{x}}_i$). Expanding the summation,

$$\mathbf{v} = v_{x_1} \hat{\mathbf{x}}_1 + \cdots + v_{x_1} \hat{\mathbf{x}}_n$$

Using matrix algebra, we can write the equation above as

$$\mathbf{v} = \begin{bmatrix} \hat{\mathbf{x}}_1 & \cdots & \hat{\mathbf{x}}_n \end{bmatrix} \underbrace{\begin{bmatrix} v_{x_1} \\ \vdots \\ v_{x_n} \end{bmatrix}}_{[\mathbf{v}]_X} \tag{1.9}$$

The vector $[\mathbf{v}]_X \in \mathbb{R}^n$ is the **coordinate vector** of $\mathbf{v}$ **expressed** or **resolved** in basis X [51, pp. 157–162], [12].

$$\boxed{[\mathbf{v}]_X = \begin{bmatrix} v_{x_1} \\ \vdots \\ v_{x_n} \end{bmatrix}} \tag{1.10}$$

> While a coordinate vector is a column vector, it is more specific than column vector. A column vector can be independent of any notion of a coordinate system, while a coordinate vector is specifically tied to a single coordinate system.

> **Convention 8:** Coordinate vector notation.
>
> A column vector expressed in an ordered basis should be placed in brackets and subscripted (outside the brackets) with the symbol representing the ordered basis. For example, a column vector, $\mathbf{v}$, expressed in the ordered basis X should be denoted
>
> $$[\mathbf{v}]_X$$
>
> If the vector has a descriptive subscript, the descriptive subscript remains inside the brackets, while the basis subscript is outside the brackets. For example, if a vector has the descriptive subscript "ex", the corresponding coordinate vector expressed in the ordered basis X is
>
> $$\left[\mathbf{v}_{\text{ex}}\right]_X$$

As an example, consider a three-dimensional column vector, $\mathbf{v} \in \mathbb{R}^3$. Let's place $\mathbf{v}$ in the three-dimensional coordinate system $Ox_1x_2x_3$ such that its tail is located at the origin. Vector $\mathbf{v}$ has components/coordinates v_1, v_2, and v_3 directed along the three coordinate axes, defined by the basis vectors $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$, respectively. For this example,
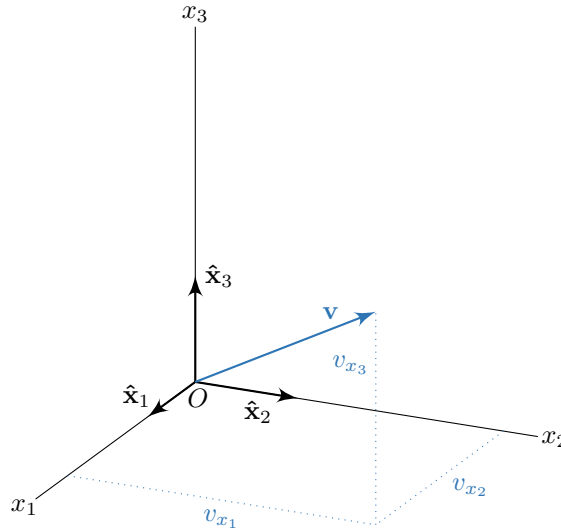
Figure 1.3: Column vector in a coordinate system.

$\mathbf{v}$ can be written in terms of the basis vectors defining $Ox_1x_2x_3$ as

$$\mathbf{v} = v_{x_1}\hat{\mathbf{x}}_1 + v_{x_2}\hat{\mathbf{x}}_2 + v_{x_3}\hat{\mathbf{x}}_3$$

and its coordinate vector expressed in X is then

$$[\mathbf{v}]_X = \begin{bmatrix} v_{x_1} \\ v_{x_2} \\ v_{x_3} \end{bmatrix}$$

1.3.1   Coordinate Basis Vectors

When we expressed a vector in a coordinate system, we obtained a coordinate vector. Similarly, when we express a basis vector in a coordinate system, we obtain a **coordinate basis vector**. This might be a bit confusing at first since basis vectors are what define coordinate systems. However, if you have multiple bases, you can express one set of

basis vectors in another basis; this will be covered more in depth in Section 1.4. As an example, consider the ordered basis

$$X = ([\hat{\mathbf{x}}_1]_Y, ..., [\hat{\mathbf{x}}_n]_Y)$$

Note that the basis vectors of X are expressed in another basis, Y.

1.3.2 The Basis Matrix

At the beginning of this section, we disregarded the basis in which basis vectors were defined when introducing the concept of a coordinate vector. Now, let's treat coordinate vectors a bit more rigorously.

Consider a coordinate system $Ox_1...x_n$ defined using the basis $X = ([\hat{\mathbf{x}}_1]_U, ..., [\hat{\mathbf{x}}_n]_U)$. Note that this time, the basis vectors of X are expressed in some other basis, $U = (\hat{\mathbf{u}}_1, ..., \hat{\mathbf{u}}_n)$. A vector expressed in U can be written as a linear combination of the basis vectors of X as

$$[\mathbf{v}]_U = \sum_{i=1}^{n} v_{x_i} [\hat{\mathbf{x}}_i]_U = v_{x_1} [\hat{\mathbf{x}}_1]_U + \cdots + v_{x_n} [\hat{\mathbf{x}}_n]_U \tag{1.11}$$

Note that $[\hat{\mathbf{x}}_i]_U$ is essentially $\hat{\mathbf{x}}_i$ projected onto $\hat{\mathbf{u}}_i$. Thus, each scalar x_i represents a coordinate with respect to the ith axis of the coordinate system defined by U.

Equivalently, we can write Eq. (1.11) using matrix algebra as

$$\mathbf{v}_U = \underbrace{\left([\hat{\mathbf{x}}_1]_U \quad \cdots \quad [\hat{\mathbf{x}}_n]_U\right)}_{[\mathbf{X}]_U} \underbrace{\begin{bmatrix} v_{x_1} \\ \vdots \\ v_{x_n} \end{bmatrix}}_{\mathbf{v}_X} \tag{1.12}$$

As before, we can easily recognize the coordinate vector $\mathbf{v}_X$. Now, let's define the **basis matrix** of X expressed in U as

$$\boxed{[\mathbf{X}]_U = \left([\hat{\mathbf{x}}_1]_U \quad \cdots \quad [\hat{\mathbf{x}}_n]_U\right)} \tag{1.13}$$

Note that since $\hat{\mathbf{x}}_i \in \mathbb{R}^n \ \forall \ i = 1, ..., n$, we have that

$$\boxed{[\mathbf{X}]_U \in \mathbb{R}^{n \times n}} \tag{1.14}$$

1.3.3 Autorepresentation

In this section, we first defined coordinate vectors in terms of "plain" basis vectors. Then, we took a step back and introduced *coordinate* basis vectors, where the basis vectors themselves were expressed in some other basis, Now, we consider the case where the basis vectors are expressed in their *own* basis (i.e. in the basis they define).

Consider the **autorepresentation** of the basis X, that is, the representation of X with respect to itself.

$$X = ([\hat{\mathbf{x}}_1]_X, ..., [\hat{\mathbf{x}}_n]_X)$$

From Eq. (1.12)

$$[\mathbf{v}]_X = \underbrace{\left([\hat{\mathbf{x}}_1]_X \quad \cdots \quad [\hat{\mathbf{x}}_n]_X\right)}_{[\mathbf{X}]_X} \underbrace{\begin{bmatrix} v_{x_1} \\ \vdots \\ v_{x_n} \end{bmatrix}}_{[\mathbf{v}]_X} \quad \rightarrow \quad [\mathbf{v}]_X = [\mathbf{X}]_X [\mathbf{v}]_X$$

Thus [39],

$$[\mathbf{X}]_X = \mathbf{I}_{n \times n} \quad \text{for the basis } X = ([\hat{\mathbf{x}}_1]_X, ..., [\hat{\mathbf{x}}_n]_X)$$

> **Convention 9:** Notation for the autorepresentation of a basis.
>
> Consider the autorepresentation of the basis X.
>
> $$X = ([\hat{\mathbf{x}}_1]_X, ..., [\hat{\mathbf{x}}_n]_X)$$
>
> To avoid cluttering notation, we write this ordered basis as
>
> $$X = (\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_n)$$
>
> In other words, if a basis is defined using coordinate basis vectors expressed in the basis itself, then we do not label the basis vectors with a basis.
>
> $$(\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_n) \equiv ([\hat{\mathbf{x}}_1]_X, ..., [\hat{\mathbf{x}}_n]_X)$$

1.4   Change of Basis

> This section is compiled from the material in [51, pp. 163–172], [92, pp. 33–34], [12], and [47].

Consider a vector $\mathbf{v} \in \mathbb{R}^n$ and two separate bases, X and Y, each with basis vectors expressed in the bases they define.

$$X = (\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_n)$$
$$Y = (\hat{\mathbf{y}}_1, ..., \hat{\mathbf{y}}_n)$$

Supposed the tail of $\mathbf{v}$ and the tails of all the basis vectors are all located at the same point, O. The basis vectors of X and Y define the coordinate systems $Ox_1...x_n$ and $Oy_1...y_n$, respectively, both with origin O.

> *Goals:*
>   1. Express $\mathbf{v}$ in the $Oy_1...y_n$ coordinate system, given its coordinates in the $Ox_1...x_n$ coordinate system. This operation represents the **change of basis** from X to Y; we obtain $[\mathbf{v}]_Y$ given $[\mathbf{v}]_X$.
>   2. Express $\mathbf{v}$ in the $Ox_1...x_n$ coordinate system, given its coordinates in the $Oy_1...y_n$ coordinate system. This operation represents the **change of basis** from Y to X; we obtain $[\mathbf{v}]_X$ given $[\mathbf{v}]_Y$.

1.4.1   Generalized Change of Basis

Recall Eq. (1.12) defining $[\mathbf{v}]_U$ in terms of the basis matrix of X with respect to basis U, where $X = ([\hat{\mathbf{x}}_1]_U, ..., [\hat{\mathbf{x}}_n]_U)$ and $U = (\hat{\mathbf{u}}_1, ..., \hat{\mathbf{u}}_n)$.

$$[\mathbf{v}]_U = \underbrace{\left([\hat{\mathbf{x}}_1]_U \quad \cdots \quad [\hat{\mathbf{x}}_n]_U\right)}_{[\mathbf{X}]_U \text{ (Eq. (1.13))}} \underbrace{\begin{bmatrix} v_{x_1} \\ \vdots \\ v_{x_n} \end{bmatrix}}_{[\mathbf{v}]_X}$$

As noted in the equation above, the first matrix is just the basis matrix of X expressed in U, while the vector is the vector $\mathbf{v}$ expressed in the basis X. Thus, we have

$$[\mathbf{v}]_U = [\mathbf{X}]_U [\mathbf{v}]_X \tag{1.15}$$
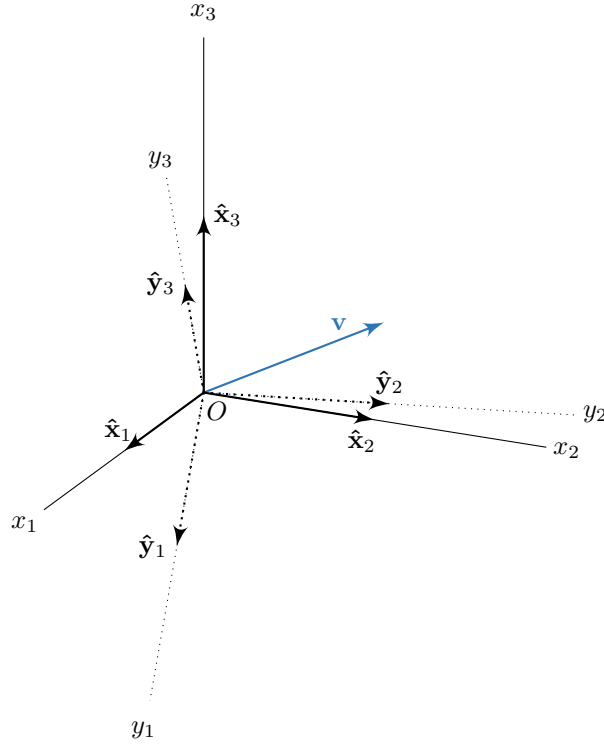
Figure 1.4: Column vector with respect to two coordinate frames.

Similarly, we could write $[\mathbf{v}]_U$ in terms of $[\mathbf{v}]_Y$ and $[\mathbf{Y}]_U$, where $Y = ([\hat{\mathbf{y}}_1]_U, ..., [\hat{\mathbf{y}}_n]_U)$ is another basis with its basis vectors expressed in U.

$$[\mathbf{v}]_U = [\mathbf{Y}]_U[\mathbf{v}]_Y \tag{1.16}$$

From Eqs. (1.15) and (1.16), we have

$$[\mathbf{X}]_U[\mathbf{v}]_X = [\mathbf{Y}]_U[\mathbf{v}]_Y \tag{1.17}$$

Solving for $[\mathbf{v}]_Y$ by left-multiplying both sides by $[\mathbf{Y}]_U^{-1}$,

$$[\mathbf{v}]_Y = [\mathbf{Y}]_U^{-1}[\mathbf{X}]_U[\mathbf{v}]_X$$

Below, we more formally define this change of basis.

Formal Definitions

Consider two unique bases, X and Y, which both have basis vectors expressed in a third basis U.

$$X = ([\hat{\mathbf{x}}_1]_U, ..., [\hat{\mathbf{x}}_n]_U)$$
$$Y = ([\hat{\mathbf{y}}_1]_U, ..., [\hat{\mathbf{y}}_n]_U)$$
$$U = (\hat{\mathbf{u}}_1, ..., \hat{\mathbf{u}}_n)$$

Suppose we know the basis vectors of X and Y expressed in U.

$$[\mathbf{X}]_U = \begin{pmatrix} [\hat{\mathbf{x}}_1]_U & \cdots & [\hat{\mathbf{x}}_n]_U \end{pmatrix}$$

$$[\mathbf{Y}]_U = \begin{pmatrix} [\hat{\mathbf{y}}_1]_U & \cdots & [\hat{\mathbf{y}}_n]_U \end{pmatrix}$$

The **generalized change of basis** from X to Y is defined as

$$\boxed{[\mathbf{v}]_Y = [\mathbf{Y}]_U^{-1}[\mathbf{X}]_U[\mathbf{v}]_X \qquad \text{(generalized change of basis } X \to Y)} \tag{1.18}$$

Similarly, the **generalized change of basis** from Y to X is defined as

$$\boxed{[\mathbf{v}]_X = [\mathbf{X}]_U^{-1}[\mathbf{Y}]_U[\mathbf{v}]_Y \qquad \text{(generalized change of basis } Y \to X)} \tag{1.19}$$

1.4.2 Standard Change of Basis

From Eq. (1.18), we know that the generalized change of basis from X to Y is

$$[\mathbf{v}]_Y = [\mathbf{Y}]_U^{-1}[\mathbf{X}]_U[\mathbf{v}]_X$$

where $U = (\hat{\mathbf{u}}_1, ..., \hat{\mathbf{u}}_n)$ is some third basis that both X and Y are defined with respect to. Now, let's consider the case where $U = Y$, which essentially means that we have the basis vectors of X expressed in the basis Y.

$$[\mathbf{v}]_Y = \underbrace{[\mathbf{Y}]_Y^{-1}}_{\mathbf{I}_{n \times n}}[\mathbf{X}]_Y[\mathbf{v}]_X$$

Note that $[\mathbf{Y}]_Y$ is the basis matrix of an autorepresentation, which we showed in Section 1.3.3 is just the identity matrix. Therefore, this change of basis reduces to

$$[\mathbf{v}]_Y = [\mathbf{X}]_Y[\mathbf{v}]_X$$

We can perform the same procedure with Eq. (1.19), setting $U = X$, to find

$$[\mathbf{v}]_X = [\mathbf{Y}]_X[\mathbf{v}]_Y$$

Below, we more formally define this change of basis.

Formal Definitions

Consider two unique bases, X and Y. Assume that we have the vector $[\mathbf{v}]_X$ expressed in X, together with the basis vectors of X expressed in Y. The basis vectors of X form the basis matrix

$$[\mathbf{X}]_Y = \begin{pmatrix} [\hat{\mathbf{x}}_1]_Y & \cdots & [\hat{\mathbf{x}}_n]_Y \end{pmatrix}$$

The **standard change of basis** from X to Y is defined as

$$\boxed{[\mathbf{v}]_Y = [\mathbf{X}]_Y[\mathbf{v}]_X \qquad \text{(standard change of basis } X \to Y)} \tag{1.20}$$

Similarly, assume that we have the vector $[\mathbf{v}]_Y$ expressed in Y, together with the basis vectors of Y expressed in X. The basis vectors of Y form the basis matrix

$$[\mathbf{Y}]_X = \begin{pmatrix} [\hat{\mathbf{y}}_1]_X & \cdots & [\hat{\mathbf{y}}_n]_X \end{pmatrix}$$

The **standard change of basis** from Y to X is defined as

$$\boxed{[\mathbf{v}]_X = [\mathbf{Y}]_X[\mathbf{v}]_Y \qquad \text{(standard change of basis } Y \to X)} \tag{1.21}$$

Note that we can also obtain $[\mathbf{v}]_X$ by left-multiplying both sides of Eq. (1.20) by $[\mathbf{X}]_Y^{-1}$.

$$[\mathbf{v}]_Y = [\mathbf{X}]_Y[\mathbf{v}]_X \quad \to \quad [\mathbf{X}]_Y^{-1}[\mathbf{v}]_Y = [\mathbf{X}]_Y^{-1}[\mathbf{X}]_Y[\mathbf{v}]_X \quad \to \quad [\mathbf{X}]_Y^{-1}[\mathbf{v}]_Y = [\mathbf{v}]_X$$

$$\therefore [\mathbf{v}]_X = [\mathbf{X}]_Y^{-1}[\mathbf{v}]_Y$$

Comparing this result with Eq. (1.21) implies that

$$\boxed{[\mathbf{Y}]_X = [\mathbf{X}]_Y^{-1}} \tag{1.22}$$

Repeating the same process but this time left-multiplying both sides of Eq. (1.21) by $[\mathbf{Y}]_X^{-1}$ and then comparing the result with Eq. (1.20) would result in

$$\boxed{[\mathbf{X}]_Y = [\mathbf{Y}]_X^{-1}} \tag{1.23}$$

1.5 Orthogonality

Orthogonal Vectors

Two vectors, $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^n$, are said to be orthogonal if their inner product is 0.

$$\boxed{\mathbf{v}_1^T \mathbf{v}_2 = 0 \quad \rightarrow \quad \mathbf{v}_1 \in \mathbb{R}^n \text{ and } \mathbf{v}_2 \in \mathbb{R}^n \text{ are orthogonal}} \tag{1.24}$$

Orthonormal Sets

An **orthonormal set** is a set of unit vectors that are all orthogonal to one another. Let $U = \{\hat{\mathbf{u}}_1, ..., \hat{\mathbf{u}}_n\}$ be a set of n vectors such that

$$\hat{\mathbf{u}}_i^T \hat{\mathbf{u}}_j = \delta_{ij}$$

where

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

Then U is an orthonormal set.

Note that the condition above implies that all the vectors are both orthogonal to one another and have magnitude 1. For the case where $i \neq j$, $\hat{\mathbf{u}}_i^T \hat{\mathbf{u}}_j = 0$ implies that the vectors are orthogonal. For the case where $i = j$, $\hat{\mathbf{u}}_i^T \hat{\mathbf{u}}_i = 1$ implies that $\hat{\mathbf{u}}_i$ is a unit vector, since $\hat{\mathbf{u}}_i^T \hat{\mathbf{u}}_i = \|\hat{\mathbf{u}}_i\|^2$.

Orthogonal Matrices

A square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is said to be **orthogonal** if its column vectors form an orthonormal set. Let $\mathbf{a}_i$ be the ith column vector of $\mathbf{A}$.

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_i & \cdots & \mathbf{a}_n \end{bmatrix}$$

$\mathbf{A}$ is an orthogonal matrix if

$$\|\mathbf{a}_i\| = 1 \,\forall\, i = 1, .., n$$
$$\mathbf{a}_i^T \mathbf{a}_j = 0 \,\forall\, i, j = 1, .., n, \ i \neq j$$

Properties of Orthogonal Matrices

Consider multiplying $\mathbf{A}$ by its transpose from the left. Writing the matrices in terms of their column vectors,

$$\mathbf{A}^T \mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{bmatrix}^T \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1^T \\ \vdots \\ \mathbf{a}_n^T \end{bmatrix} \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1^T \mathbf{a}_1 & \mathbf{a}_1^T \mathbf{a}_2 & \cdots & \mathbf{a}_1^T \mathbf{a}_n \\ \mathbf{a}_2^T \mathbf{a}_1 & \mathbf{a}_2^T \mathbf{a}_2 & \cdots & \mathbf{a}_2^T \mathbf{a}_n \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_n^T \mathbf{a}_1 & \mathbf{a}_n^T \mathbf{a}_2 & \cdots & \mathbf{a}_n^T \mathbf{a}_n \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

It follows that an orthogonal matrix has the property

$$\boxed{\mathbf{A}^T \mathbf{A} = \mathbf{A} \mathbf{A}^T = \mathbf{I}_{n \times n}} \tag{1.25}$$

where $\mathbf{I}_{n \times n}$ is the $n \times n$ identity matrix.

We also know that $\mathbf{A}^{-1}\mathbf{A} = \mathbf{A}\mathbf{A}^{-1} = \mathbf{I}_{n \times n}$, which when compared to Eq. (1.25) yields the property

$$\boxed{\mathbf{A}^{-1} = \mathbf{A}^T} \tag{1.26}$$

Eq. (1.26) is an extremely important property because it is computationally much more efficient to calculate the transpose of a matrix than its inverse.

Next, consider the determinant of $\mathbf{A}\mathbf{A}^T$. We know that

$$\det\left(\mathbf{I}_{n \times n}\right) = 1$$

By the definition of the determinant, we know that

$$\det\left(\mathbf{A}\mathbf{A}^T\right) = \det\left(\mathbf{A}\right)\det\left(\mathbf{A}^T\right)$$

Since $\det\left(\mathbf{A}^T\right) = \det\left(\mathbf{A}\right)$ for any square matrix $\mathbf{A}$,

$$\det\left(\mathbf{A}\mathbf{A}^T\right) = \det\left(\mathbf{A}\right)^2$$

Since $\mathbf{A}\mathbf{A}^T = \mathbf{I}_{n \times n}$,

$$\det\left(\mathbf{I}_{n \times n}\right) = \det\left(\mathbf{A}\right)^2 \quad \rightarrow \quad \det\left(\mathbf{A}\right)^2 = 1$$

$$\boxed{\left|\det\left(\mathbf{A}\right)\right| = 1} \tag{1.27}$$

In Eq. (1.27), $|\cdot|$ denotes an absolute value while $\det\left(\cdot\right)$ denotes a determinant; this is essential to note because $|\cdot|$ is often used to denote the determinant [85].

> If a matrix is orthogonal, its determinant is either -1 or 1. ***However***, if the determinant of a square matrix is -1 or 1, it does not imply that the matrix is orthogonal [27].

Now, consider two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and the orthogonal matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$. Taking the inner product of $\mathbf{A}\mathbf{x}$ with $\mathbf{A}\mathbf{y}$,

$$(\mathbf{A}\mathbf{x})^T(\mathbf{A}\mathbf{y}) = \mathbf{x}^T \underbrace{\mathbf{A}^T\mathbf{A}}_{\mathbf{I}_{n \times n}} \mathbf{y}$$

$$\boxed{(\mathbf{A}\mathbf{x})^T(\mathbf{A}\mathbf{y}) = \mathbf{x}^T\mathbf{y}} \tag{1.28}$$

In the case that $\mathbf{x} = \mathbf{y}$,

$$(\mathbf{A}\mathbf{x})^T\mathbf{A}\mathbf{x} = \mathbf{x}^T\mathbf{x} = \|\mathbf{x}\|_2^2$$

We also know that

$$(\mathbf{A}\mathbf{x})^T(\mathbf{A}\mathbf{x}) = \|\mathbf{A}\mathbf{x}\|^2$$

Thus, we have [51, pp. 263–268][95, p. 15]

$$\boxed{\|\mathbf{A}\mathbf{x}\| = \|\mathbf{x}\|} \tag{1.29}$$

Product of Orthogonal Matrices

Consider two orthogonal matrices $\mathbf{A}$ and $\mathbf{B}$. Their product is given by $\mathbf{A}\mathbf{B}$. Thus, from Eq. (1.26), we know that if $(\mathbf{A}\mathbf{B})^T(\mathbf{A}\mathbf{B}) = \mathbf{I}$, then the matrix product $\mathbf{A}\mathbf{B}$ is also orthogonal.

$$(\mathbf{A}\mathbf{B})^T(\mathbf{A}\mathbf{B}) = \left(\mathbf{B}^T\mathbf{A}^T\right)(\mathbf{A}\mathbf{B}) = \mathbf{B}^T\left(\mathbf{A}^T\mathbf{A}\right)\mathbf{B} = \mathbf{B}^T\mathbf{I}\mathbf{B} = \mathbf{B}^T\mathbf{B} = \mathbf{I}$$

Thus, the product of two orthogonal matrices is also an orthogonal matrix [107].

1.5.1   Orthonormal Bases

Consider a ordered basis $X_U = ([\hat{\mathbf{x}}_1]_U, ..., [\hat{\mathbf{x}}_n]_U)$ with basis vectors expressed in another basis U with a corresponding basis matrix

$$[\mathbf{X}]_U = \begin{pmatrix} [\hat{\mathbf{x}}_1]_U & \cdots & [\hat{\mathbf{x}}_n]_U \end{pmatrix}$$

If $[\mathbf{X}]_U$ is an orthogonal matrix, or equivalently, if all the basis vectors defining X are mutually orthogonal, then X is an **orthonormal basis**.

Let's consider the case where X and Y are both orthonormal bases defining coordinate systems, with corresponding coordinate basis matrices

$$[\mathbf{X}]_U = \begin{pmatrix} [\hat{\mathbf{x}}_1]_U & \cdots & [\hat{\mathbf{x}}_n]_U \end{pmatrix}, \qquad [\mathbf{Y}]_U = \begin{pmatrix} [\hat{\mathbf{y}}_1]_U & \cdots & [\hat{\mathbf{y}}_n]_U \end{pmatrix}$$

Since the basis vectors of X and Y both form orthonormal sets, $[\mathbf{X}]_U$ and $[\mathbf{Y}]_U$ are orthogonal matrices, implying (from Eq. (1.26)) that $[\mathbf{X}]_U^{-1} = [\mathbf{X}]_U^T$ and $[\mathbf{Y}]_U^{-1} = [\mathbf{Y}]_U^T$. The generalized change of basis formulas from Section 1.4.1 can then be simplified to

$$\boxed{[\mathbf{v}]_Y = [\mathbf{Y}]_U^T [\mathbf{X}]_U [\mathbf{v}]_X} \qquad \text{(generalized change of basis } X \to Y \text{ where } Y \text{ is an orthonormal basis)} \qquad (1.30)$$

$$\boxed{[\mathbf{v}]_X = [\mathbf{X}]_U^T [\mathbf{Y}]_U [\mathbf{v}]_Y} \qquad \text{(generalized change of basis } Y \to X \text{ where } X \text{ is an orthonormal basis)} \qquad (1.31)$$

Similarly, we can replace the inverses in Eqs. (1.22) and (1.23) to obtain

$$\boxed{[\mathbf{Y}]_X = [\mathbf{X}]_Y^T} \quad \text{(if } X \text{ is an orthonormal basis)} \qquad (1.32)$$

$$\boxed{[\mathbf{X}]_Y = [\mathbf{Y}]_X^T} \quad \text{(if } Y \text{ is an orthonormal basis)} \qquad (1.33)$$

Note that these definitions are much faster computationally since transposing a matrix requires much fewer operations than performing a matrix inversion.

1.6   Cartesian Coordinate Systems and Euclidean Spaces

The **Euclidean space** is the fundamental space of geometry. More abstractly, Euclidean space can be thought of in a coordinate-free sense (similar to how reference frames are coordinate free; see Section 2.2) as representing physical space. Points in an n-dimensional Euclidean space can be defined using vectors in the real n-space $\mathbb{R}^n$. These vectors can be expressed with respect to *any* basis that spans $\mathbb{R}^n$.

A specific subset of bases that span $\mathbb{R}^n$ are n-dimensional orthonormal basis. A **Cartesian coordinate system** is simply a coordinate system defined by an orthonormal basis. Like any coordinate system with basis spanning $\mathbb{R}^n$, an n-dimensional Cartesian coordinate system can be used to specify points in an n-dimensional Euclidean space.[20, 32].

A special case of great importance is the **Euclidean 3-space**, which can be described using 3D Cartesian coordinate systems. The Euclidean 3-space is fundamental to the study of kinematics and dynamics since our universe has three physical dimensions [97].

1.7   Rotation Matrices

Recall from Section **??** that a Cartesian coordinate system is defined by an orthonormal basis. While we already described the generalized change of basis for n-dimensional Cartesian coordinate systems, there are special ways we can perform a change of basis in a Euclidean 3-space. In a Euclidean 3-space, the change of basis from one Cartesian coordinate system to another can be represented using rotations.

1.7.1 Passive vs. Active Rotations

When dealing with rotations of coordinate spaces and vectors, there are two common ways that rotations are performed:

1. **Passive Rotation:** The original coordinate system is rotated by some angle θ about one of its axes, while any vector or matrix quantities remain constant, i.e. *passive*, with respect to the original coordinate systems. The vector and matrix quantities are then expressed in the new coordinate system.
2. **Active Rotation:** The coordinate system remains stationary and the vector or matrix *actively* rotates with respect to the original coordinate system.

In fields such as computer graphics and video games, active rotations are generally used because there is usually some object moving in a fixed coordinate system. However, in the field of dynamics, specifically aerospace dynamics, we typically use passive rotations, since we will have a vector that we want to *resolve* or *express* in different coordinate systems [2].

> **Convention 10:** Use of passive rotations.
>
> In dynamics, we use **passive rotations** to transform the coordinates of a vector from one coordinate system to another.

1.7.2 Elementary Rotations

Consider the $Ox_1x_2x_3$ coordinate system. We refer to the three axes of the this coordinate system as either the **first**, **second**, or **third axes**.

> **Convention 11:** Numbering axes.
>
> 1. $x_1 = $ 1st axis
> 2. $x_2 = $ 2nd axis
> 3. $x_3 = $ 3rd axis

We define the **rotation matrix**, or more explicitly, the **passive rotation matrix**, for a counterclockwise rotation of θ about the ith axis of a coordinate system as $\mathbf{R}_i(\theta)$. There are three **elementary rotations**, each about a different axes, encoded by $\mathbf{R}_1$, $\mathbf{R}_2$, and $\mathbf{R}_3$, respectively:

1. $\mathbf{R}_1(\theta)$: Encodes counterclockwise rotation by θ of the 2nd and 3rd axes (x_2x_3-plane) about the 1st axis (x_1).
2. $\mathbf{R}_2(\theta)$: Encodes counterclockwise rotation by θ of the 1st and 3rd axes (x_1x_3-plane) about the 2nd axis (x_2).
3. $\mathbf{R}_3(\theta)$: Encodes counterclockwise rotation by θ of the 1st and 2nd axes (x_1x_2-plane) about the 3rd axis (x_3).

These rotations are illustrated in Fig. 1.5. The (passive) rotation matrices $\mathbf{R}_1(\theta)$, $\mathbf{R}_2(\theta)$, and $\mathbf{R}_3(\theta)$ are defined by Eqs. (1.34), (1.35), and (1.36), respectively [103, p. 162], [33], [89].

$$\mathbf{R}_1(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \tag{1.34}$$

$$\mathbf{R}_2(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{1.35}$$

(a) Rotation about 1st axis.

(b) Rotation about 2nd axis.

(c) Rotation about 3rd axis.
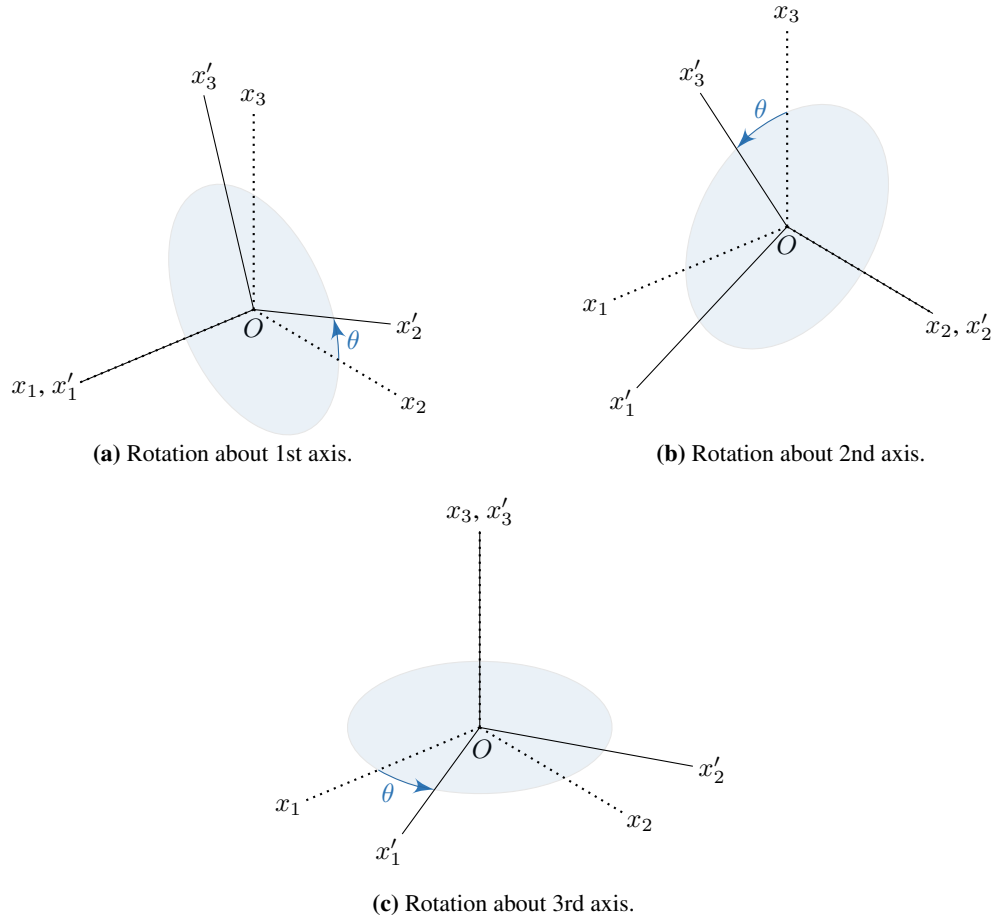
Figure 1.5: Elementary rotations.

$$\mathbf{R}_3(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.36}$$

We formalize these simple equations as algorithms since they will be used often in other algorithms.

Algorithm 1: `rot1`

Rotation matrix for a passive rotation about the 1st axis.

Inputs:

- $\theta \in \mathbb{R}$ - angle of rotation $[\mathrm{rad}]$

Procedure:

1. Precompute the trigonometric functions.

$$c = \cos\theta$$
$$s = \sin\theta$$

2. Construct the rotation matrix.

$$\mathbf{R}_1(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & s \\ 0 & -s & c \end{bmatrix}$$

Outputs:
- $\mathbf{R}_1(\theta) \in \mathbb{R}^{3\times3}$ - passive rotation matrix about 1st axis

Test Cases:
- See Appendix A.1.1.

Algorithm 2: `rot2`

Rotation matrix for a passive rotation about the 2nd axis.

Inputs:
- $\theta \in \mathbb{R}$ - angle of rotation $[\mathrm{rad}]$

Procedure:
1. Precompute the trigonometric functions.

$$c = \cos\theta$$
$$s = \sin\theta$$

2. Construct the rotation matrix.

$$\mathbf{R}_2(\theta) = \begin{bmatrix} c & 0 & -s \\ 0 & 1 & 0 \\ s & 0 & c \end{bmatrix}$$

Outputs:
- $\mathbf{R}_2(\theta) \in \mathbb{R}^{3\times3}$ - passive rotation matrix about 2nd axis

Test Cases:
- See Appendix A.1.1.

Algorithm 3: `rot3`

Rotation matrix for a passive rotation about the 3rd axis.

Inputs:
- $\theta \in \mathbb{R}$ - angle of rotation $[\mathrm{rad}]$

Procedure:
1. Precompute the trigonometric functions.

$$c = \cos\theta$$
$$s = \sin\theta$$

2. Construct the rotation matrix.

$$\mathbf{R}_3(\theta) = \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Outputs:
- $\mathbf{R}_3(\theta) \in \mathbb{R}^{3\times3}$ - passive rotation matrix about 3rd axis

Test Cases:
- See Appendix A.1.1.

1.7.3 Properties of Elementary Rotation Matrices

Let $X = (\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{\mathbf{x}}_3)$ and $X' = (\hat{\mathbf{x}}_1', \hat{\mathbf{x}}_2', \hat{\mathbf{x}}_3')$ be two ordered bases that are both orthonormal bases, where X' is defined by rotating X about the x_3 axis by an angle θ. Consider a vector, $[\mathbf{v}]_X$, expressed in the coordinate system defined by X. To resolve that same vector in the coordinate system defined by X', we could perform the change of basis as defined by Eq. (1.20).

$$[\mathbf{v}]_{X'} = \mathbf{X}_{X'}[\mathbf{v}]_X$$

However, we can note that this same change of basis can be performed by the elementary rotation matrix $\mathbf{R}_3(\theta)$.

$$[\mathbf{v}]_{X'} = \mathbf{R}_3(\theta)[\mathbf{v}]_X$$

Since $\mathbf{X}_{X'}$ is orthogonal (X is an orthonormal basis), and since $\mathbf{R}_3(\theta) = \mathbf{X}_{X'}$, we know that $\mathbf{R}_3(\theta)$ is orthogonal. In general,

Elementary rotation matrices, $\mathbf{R}_i(\theta)$, are orthogonal matrices.

Consider defining an elementary rotation matrix using a negative angle.

$$\mathbf{R}_1(-\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(-\theta) & \sin(-\theta) \\ 0 & -\sin(-\theta) & \cos(-\theta) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}^T = \mathbf{R}_1(\theta)^T$$

$$\mathbf{R}_2(-\theta) = \begin{bmatrix} \cos(-\theta) & 0 & -\sin(-\theta) \\ 0 & 1 & 0 \\ \sin(-\theta) & 0 & \cos(-\theta) \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}^T = \mathbf{R}_2(\theta)^T$$

$$\mathbf{R}_3(-\theta) = \begin{bmatrix} \cos(-\theta) & \sin(-\theta) & 0 \\ -\sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}^T = \mathbf{R}_3(\theta)^T$$

Thus, in general, we have

$$\mathbf{R}_i(-\theta) = \mathbf{R}_i(\theta)^T$$

However, we also know that since $\mathbf{R}_i(\theta)$ is orthogonal, its inverse is equal to its transpose. Therefore, we have

$$\boxed{\mathbf{R}_i(\theta)^{-1} = \mathbf{R}_i(\theta)^T = \mathbf{R}_i(-\theta)} \tag{1.37}$$

Since elementary rotation matrices are orthogonal, they also share all the other properties of orthogonal matrices as summarized in Section 1.5. Note that in the properties below, $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$.

$$\boxed{|\det[\mathbf{R}_i(\theta)]| = 1} \tag{1.38}$$

$$\boxed{\mathbf{R}_i(\theta)^T \mathbf{R}_i(\theta) = \mathbf{R}_i(\theta)\mathbf{R}_i(\theta)^T = \mathbf{I}_{3\times 3}} \tag{1.39}$$

$$\boxed{\left[\mathbf{R}_i(\theta)\mathbf{x}\right]^T \left[\mathbf{R}_i(\theta)\mathbf{y}\right] = \mathbf{x}^T\mathbf{y}} \tag{1.40}$$

$$\boxed{\|\mathbf{R}_i(\theta)\mathbf{x}\| = \|\mathbf{x}\|} \tag{1.41}$$

Since $\mathbf{R}_i(-\theta) = \mathbf{R}_i(\theta)^T$, we can also get an alternate version of the property presented by Eq. (1.39).

$$\boxed{\mathbf{R}_i(\theta)\mathbf{R}_i(-\theta) = \mathbf{I}_{3\times 3}} \tag{1.42}$$

1.7.4   Sequential Rotations

Consider an arbitrary vector $\mathbf{v} \in \mathbb{R}^3$ expressed with respect to the ordered basis $S = (\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$, where S defines the coordinate system $Oxyz$.

$$\mathbf{v}_S = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$$

First, rotating S by θ_1 about the 1st axis (x), we obtain a new coordinate system $Ox'y'z'$ defined by the ordered basis $S' = (\hat{\mathbf{x}}', \hat{\mathbf{y}}', \hat{\mathbf{z}}')$. Expressing $\mathbf{v}$ in the $Ox'y'z'$ coordinate system (i.e. with respect to the ordered basis S'),

$$\mathbf{v}_{S'} = \mathbf{R}_1(\theta_1)\mathbf{v}_S$$

Next, rotating S' by θ_2 about the 2nd axis (y'), we obtain a third coordinate system, $Ox''y''z''$ defined by the ordered basis $S'' = (\hat{\mathbf{x}}'', \hat{\mathbf{y}}'', \hat{\mathbf{z}}'')$. Expressing $\mathbf{v}$ in the $Ox''y''z''$ coordinate system (i.e. with respect to the ordered basis S''),

$$\begin{aligned} \mathbf{v}_{S''} &= \mathbf{R}_2(\theta_2)\mathbf{v}_{S'} \\ &= \mathbf{R}_2(\theta_2)\mathbf{R}_1(\theta_1)\mathbf{v}_S \end{aligned}$$

In general, for a passive rotation sequence $i \to j \to k$ (where i, j, and k can be either 1, 2, or 3, i.e. representing an axis), we define

$$\boxed{\mathbf{R}_{ijk}(\theta_1, \theta_2, \theta_3) = \mathbf{R}_k(\theta_3)\mathbf{R}_j(\theta_2)\mathbf{R}_i(\theta_1) \qquad \text{(for the passive rotation sequence } i \to j \to k\text{)}} \tag{1.43}$$

> **Convention 12:** Sequential passive rotation.
>
> The i-j-k passive rotation sequence $(i \to j \to k)$ is described by the passive rotation matrix
> $$\mathbf{R}_{ijk}(\theta_1, \theta_2, \theta_3)$$
> where
> 1. The angles are input in the order the rotations are applied, so their subscript corresponds to the current rotation. For example, θ_2 is used for the second rotation.
> 2. All angles assumes counterclockwise is positive.
> 3. The rotations are applied in the order of the subscript:
>   (a) First rotation is about the ith axis.
>   (b) Second rotation is about the jth axis.
>   (c) Third rotation is about the kth axis.

There are two specific passive rotation sequences that are typically used in aerospace simulation. The two sequences are described in detail below.

3-2-1 rotation sequence

The 3-2-1 $(3 \to 2 \to 1)$ rotation sequence consists of the following steps:

1. rotation of θ_1 about z (3rd axis)
2. rotation of θ_2 about y' (2nd axis)
3. rotation of θ_3 about x'' (1st axis)

The passive rotation matrix for the 3-2-1 sequence is [33][2], [105, pp. 763-764][3]

$$\mathbf{R}_{321}(\theta_1, \theta_2, \theta_3) = \mathbf{R}_1(\theta_3)\mathbf{R}_2(\theta_2)\mathbf{R}_3(\theta_1)$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_3 & \sin\theta_3 \\ 0 & -\sin\theta_3 & \cos\theta_3 \end{bmatrix} \begin{bmatrix} \cos\theta_2 & 0 & -\sin\theta_2 \\ 0 & 1 & 0 \\ \sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix} \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & 0 \\ -\sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_{321}(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} \cos\theta_2\cos\theta_1 & \cos\theta_2\sin\theta_1 & -\sin\theta_2 \\ -\cos\theta_3\sin\theta_1 + \sin\theta_3\sin\theta_2\cos\theta_1 & \cos\theta_3\cos\theta_1 + \sin\theta_3\sin\theta_2\sin\theta_1 & \sin\theta_3\cos\theta_2 \\ \sin\theta_3\sin\theta_1 + \cos\theta_3\sin\theta_2\cos\theta_1 & -\sin\theta_3\cos\theta_1 + \cos\theta_3\sin\theta_2\sin\theta_1 & \cos\theta_3\cos\theta_2 \end{bmatrix}$$

$$(1.44)$$

> See B.1 for a computational derivation of Eq. (1.44).

We formalize Eq. (1.44) as Algorithm 4 below. Note that in its implementation, we precompute the trigonometric functions *before* populating the matrix to decrease the computational cost.

Algorithm 4: `rot321`
Rotation matrix for the 3-2-1 rotation sequence.

Inputs:
- $\theta_1 \in \mathbb{R}$ - angle for first rotation (about 3rd axis) [rad]
- $\theta_2 \in \mathbb{R}$ - angle for second rotation (about 2nd axis) [rad]
- $\theta_3 \in \mathbb{R}$ - angle for third rotation (about 1st axis) [rad]

Procedure:
1. Precompute the trigonometric functions.

$$s_1 = \sin\theta_1$$
$$c_1 = \cos\theta_1$$
$$s_2 = \sin\theta_2$$
$$c_2 = \cos\theta_2$$
$$s_3 = \sin\theta_3$$
$$c_3 = \cos\theta_3$$

2. Construct the rotation matrix.

$$\mathbf{R}_{321}(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} c_2 c_1 & c_2 s_1 & -s_2 \\ -c_3 s_1 + s_3 s_2 c_1 & c_3 c_1 + s_3 s_2 s_1 & s_3 c_2 \\ s_3 s_1 + c_3 s_2 c_1 & -s_3 c_1 + c_3 s_2 s_1 & c_3 c_2 \end{bmatrix}$$

Outputs:
- $\mathbf{R}_{321}(\theta_1, \theta_2, \theta_3) \in \mathbb{R}^{3\times 3}$ - passive rotation matrix for 3-2-1 rotation sequence

---

[2] In this reference, $\theta_1 = \psi$, $\theta_2 = \theta$, and $\theta_3 = \phi$.
[3] In this reference, $\theta_1 = \phi$, $\theta_2 = \theta$, and $\theta_3 = \psi$.

Test Cases:
- See Appendix A.1.1.

3-1-3 rotation sequence

The 3-1-3 ($3 \to 1 \to 3$) rotation sequence consists of the following steps:
1. rotation of θ_1 about x_3 (3rd axis)
2. rotation of θ_2 about x'_1 (1st axis)
3. rotation of θ_3 about x''_3 (3rd axis)

The rotation matrix for the 3-1-3 sequence is [33], [105, pp. 763-764][4]

$$\mathbf{R}_{313}(\theta_1, \theta_2, \theta_3) = \mathbf{R}_3(\theta_3)\mathbf{R}_1(\theta_2)\mathbf{R}_3(\theta_1)$$

$$= \begin{bmatrix} \cos\theta_3 & \sin\theta_3 & 0 \\ -\sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_2 & \sin\theta_2 \\ 0 & -\sin\theta_2 & \cos\theta_2 \end{bmatrix} \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & 0 \\ -\sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_{313}(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} \cos\theta_3\cos\theta_1 - \sin\theta_3\cos\theta_2\sin\theta_1 & \cos\theta_3\sin\theta_1 + \sin\theta_3\cos\theta_2\cos\theta_1 & \sin\theta_3\sin\theta_2 \\ -\sin\theta_3\cos\theta_1 - \cos\theta_3\cos\theta_2\sin\theta_1 & -\sin\theta_3\sin\theta_1 + \cos\theta_3\cos\theta_2\cos\theta_1 & \cos\theta_3\sin\theta_2 \\ \sin\theta_2\sin\theta_1 & -\sin\theta_2\cos\theta_1 & \cos\theta_2 \end{bmatrix}$$

$$(1.45)$$

> See B.2 for a computational derivation of Eq. (1.45).

We formalize Eq. (1.45) as Algorithm 5 below. Note that in its implementation, we precompute the trigonometric functions *before* populating the matrix to decrease the computational cost.

Algorithm 5: `rot313`
Rotation matrix for the 3-1-3 rotation sequence.

Inputs:
- $\theta_1 \in \mathbb{R}$ - angle for first rotation (about 3rd axis) [rad]
- $\theta_2 \in \mathbb{R}$ - angle for second rotation (about 1st axis) [rad]
- $\theta_3 \in \mathbb{R}$ - angle for third rotation (about 3rd axis) [rad]

bigbreak

Procedure:
1. Precompute the trigonometric functions.

$$s_1 = \sin\theta_1$$
$$c_1 = \cos\theta_1$$
$$s_2 = \sin\theta_2$$
$$c_2 = \cos\theta_2$$
$$s_3 = \sin\theta_3$$
$$c_3 = \cos\theta_3$$

---

[4] In this reference, $\theta_1 = \phi$, $\theta_2 = \theta$, and $\theta_3 = \psi$.

2. Construct the rotation matrix.

$$\mathbf{R}_{313}(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} c_3 c_1 - s_3 c_2 s_1 & c_3 s_1 + s_3 c_2 c_1 & s_3 s_2 \\ -s_3 c_1 - c_3 c_2 s_1 & -s_3 s_1 + c_3 c_2 c_1 & c_3 s_2 \\ s_2 s_1 & -s_2 c_1 & c_2 \end{bmatrix}$$

Outputs:
- $\mathbf{R}_{313}(\theta_1, \theta_2, \theta_3) \in \mathbb{R}^{3 \times 3}$ - passive rotation matrix for 3-1-3 rotation sequence

1.7.5 Properties of Sequential Rotation Matrices

A sequential rotation matrix is defined as

$$\mathbf{R}_{ijk}(\theta_1, \theta_2, \theta_3) = \mathbf{R}_k(\theta_3)\mathbf{R}_j(\theta_2)\mathbf{R}_i(\theta_1)$$

$\mathbf{R}_i(\theta_1)$, $\mathbf{R}_j(\theta_2)$, and $\mathbf{R}_k(\theta_3)$ are all elementary rotation matrices, and in Section **??** we showed that elementary rotation matrices are orthogonal matrices. Additionally, from Section 1.5, we know that taking the product of orthogonal matrices results in an orthogonal matrix. Thus, we know that

Sequential rotation matrices, $\mathbf{R}_{ijk}(\theta_1, \theta_2, \theta_3)$, are orthogonal matrices.

Consider taking the transpose of $\mathbf{R}_{ijk}(\theta_1, \theta_2, \theta_3)$.

$$\mathbf{R}_{ijk}(\theta_1, \theta_2, \theta_3)^T = [\mathbf{R}_k(\theta_3)\mathbf{R}_j(\theta_2)\mathbf{R}_i(\theta_1)]^T = \mathbf{R}_i(\theta_1)^T \mathbf{R}_j(\theta_2)^T \mathbf{R}_k(\theta_3)^T = \mathbf{R}_i(-\theta_1)\mathbf{R}_j(-\theta_2)\mathbf{R}_k(-\theta_3)$$
$$= \mathbf{R}_{kji}(-\theta_3, -\theta_2, -\theta_1)$$

Since sequential rotation matrices are orthogonal matrices, their transpose is also equal to their inverse. Thus,

$$\mathbf{R}_{ijk}(\theta_1, \theta_2, \theta_3)^{-1} = \mathbf{R}_{ijk}(\theta_1, \theta_2, \theta_3)^T = \mathbf{R}_{kji}(-\theta_3, -\theta_2, -\theta_1) = \mathbf{R}_i(-\theta_1)\mathbf{R}_j(-\theta_2)\mathbf{R}_k(-\theta_3) \qquad (1.46)$$

1.8 Skew Symmetry

In Section 1.5, we covered orthogonal matrices in-depth. Another type of matrix that arises often in dynamics is the **skew-symmetrix matrix**. A skew-symmetric matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ satisfies the property

$$\mathbf{A}^T = -\mathbf{A} \qquad (1.47)$$

which also implies that

$$\mathbf{A} + \mathbf{A}^T = \mathbf{0}_{n \times n} \qquad (1.48)$$

In the context of aerospace dynamics, the special case where $\mathbf{A} \in \mathbb{R}^{3 \times 3}$ is the one of greatest importance. In the 3×3 case, all skew-symmetric matrices are of the form [87]

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 0 & -a_{21} & a_{13} \\ a_{21} & 0 & -a_{32} \\ -a_{13} & a_{32} & 0 \end{bmatrix} \qquad (1.49)$$

We can verify that $\mathbf{A} + \mathbf{A}^T = \mathbf{0}_{3\times3}$.

$$\mathbf{A} + \mathbf{A}^T = \begin{bmatrix} 0 & -a_{21} & a_{13} \\ a_{21} & 0 & -a_{32} \\ -a_{13} & a_{32} & 0 \end{bmatrix} + \begin{bmatrix} 0 & -a_{21} & a_{13} \\ a_{21} & 0 & -a_{32} \\ -a_{13} & a_{32} & 0 \end{bmatrix}^T$$

$$= \begin{bmatrix} 0 & -a_{21} & a_{13} \\ a_{21} & 0 & -a_{32} \\ -a_{13} & a_{32} & 0 \end{bmatrix} + \begin{bmatrix} 0 & a_{21} & -a_{13} \\ -a_{21} & 0 & a_{32} \\ a_{13} & -a_{32} & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$= \mathbf{0}_{3\times3}$$

1.9 Vector Operations and Properties in 3D Cartesian Coordinate Systems

Let $Oxyz$ be a 3D Cartesian coordinate system defined by the orthonormal ordered basis $A = (\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$. Now, consider column vectors (illustrated in Fig. 1.6), $\mathbf{a}$ and $\mathbf{b}$, which can be written as linear combinations of the basis vectors of E.

$$\mathbf{a} = a_x\hat{\mathbf{x}} + a_y\hat{\mathbf{y}} + a_z\hat{\mathbf{z}}$$
$$\mathbf{b} = b_x\hat{\mathbf{x}} + b_y\hat{\mathbf{y}} + b_z\hat{\mathbf{z}}$$

Below, we compile many properties of vectors in Cartesian coordinate systems [93]
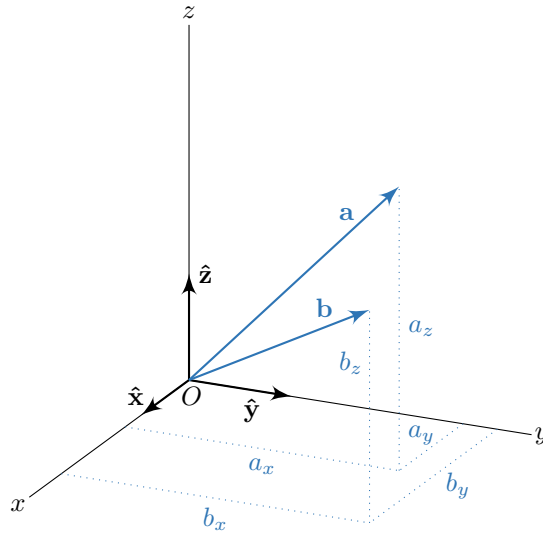
Figure 1.6: Vectors in a Cartesian coordinate system.

$\mathbf{a}$ and $\mathbf{b}$ are technically coordinate vectors since they are expressed in a coordinate system. However, since they are both expressed in the *same* coordinate system, it is not important to include the subscript labeling the frame the coordinate vectors are expressed in.

Vector addition

To add two vectors, we can simply add the individual components.

$$\mathbf{a} + \mathbf{b} = \begin{bmatrix} a_x + b_x \\ a_y + b_y \\ a_z + b_z \end{bmatrix} \tag{1.50}$$
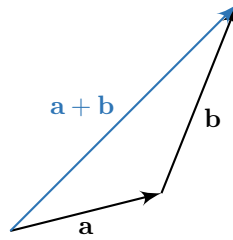
Figure 1.7: Vector addition.

Vector subtraction

To find the difference between two vectors, we can simply find the differences between the individual components.

$$\mathbf{a} - \mathbf{b} = \begin{bmatrix} a_x - b_x \\ a_y - b_y \\ a_z - b_z \end{bmatrix} \tag{1.51}$$
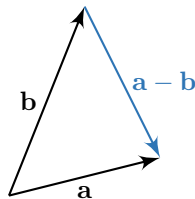
Figure 1.8: Vector subtraction.

Dot (scalar) product

The **dot (scalar) product** can be written either in terms of the components of **a** and **b**, or in terms of their magnitudes and the angle between them.

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T \mathbf{b} = a_x b_x + a_y b_y + a_z b_z = \|\mathbf{a}\| \, \|\mathbf{b}\| \cos \theta \tag{1.52}$$

The dot product also exhibits the following properties (where $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ are vectors and k is an arbitrary constant).

$$\mathbf{a} \cdot \mathbf{a} = \|\mathbf{a}\|^2 = a^2 \tag{1.53a}$$

$$\mathbf{a} \cdot (\mathbf{b} + \mathbf{c}) = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \cdot \mathbf{c} \tag{1.53b}$$

$$\mathbf{0} \cdot \mathbf{a} = 0 \tag{1.53c}$$

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{b} \cdot \mathbf{a} \tag{1.53d}$$

$$(k\mathbf{a}) \cdot \mathbf{b} = k(\mathbf{a} \cdot \mathbf{b}) = \mathbf{a} \cdot (k\mathbf{b}) \tag{1.53e}$$

Finally, we can also find the following identities for the dot products between the basis vectors $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$.

$$\hat{\mathbf{x}} \cdot \hat{\mathbf{y}} = \hat{\mathbf{y}} \cdot \hat{\mathbf{z}} = \hat{\mathbf{z}} \cdot \hat{\mathbf{x}} = 0 \tag{1.54a}$$

$$\hat{\mathbf{x}} \cdot \hat{\mathbf{x}} = \hat{\mathbf{y}} \cdot \hat{\mathbf{y}} = \hat{\mathbf{z}} \cdot \hat{\mathbf{z}} = 1 \tag{1.54b}$$

Vector magnitude

The magnitude of a vector $\mathbf{a}$ can be denoted as either a or $\|\mathbf{a}\|$ and is found by taking the 2-norm of $\mathbf{a}$. Note that instead of using $\|\cdot\|_2$ to denote the 2-norm, we just use $\|\cdot\|$. This is customary in the study of vectors in $\mathbb{R}^3$.

$$a = \|\mathbf{a}\| = \sqrt{\mathbf{a} \cdot \mathbf{a}} = \sqrt{\mathbf{a}^T \mathbf{a}} = \sqrt{a_x^2 + a_y^2 + a_z^2} \tag{1.55}$$

Unit vector in direction of $\mathbf{a}$

The **unit vector** $\hat{\mathbf{a}}$ corresponding to the vector $\mathbf{a}$ is a vector of magnitude 1 in the direction of $\mathbf{a}$.

$$\hat{\mathbf{a}} = \frac{\mathbf{a}}{\|\mathbf{a}\|} = \frac{\mathbf{a}}{a} = \frac{\mathbf{a}}{\sqrt{a_x^2 + a_y^2 + a_z^2}} \tag{1.56}$$

Cross (vector) product

The **cross (vector) product** of $\mathbf{a}$ and $\mathbf{b}$ is a vector multiplication that produces a third vector that is normal/orthogonal/perpendicular to both $\mathbf{a}$ and $\mathbf{b}$.

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} b_z a_y - a_z b_y \\ b_x a_z - b_z a_x \\ b_y a_x - b_x a_y \end{bmatrix} \tag{1.57}$$

$$\|\mathbf{a} \times \mathbf{b}\| = \|\mathbf{a}\| \, \|\mathbf{b}\| \sin\theta \tag{1.58}$$

The cross product also exhibits the following properties (where $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ are vectors and k is an arbitrary constant).
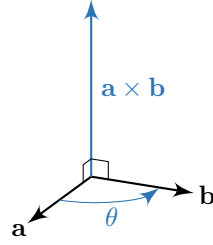
Figure 1.9: Cross product.

$$\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a} \tag{1.59a}$$

$$(k\mathbf{a}) \times \mathbf{b} = k(\mathbf{a} \times \mathbf{b}) = \mathbf{a} \times (k\mathbf{b}) \tag{1.59b}$$

$$\mathbf{a} \times (\mathbf{b} + \mathbf{c}) = \mathbf{a} \times \mathbf{b} + \mathbf{a} \times \mathbf{c} \tag{1.59c}$$

$$(\mathbf{a} + \mathbf{b}) \times \mathbf{c} = \mathbf{a} \times \mathbf{c} + \mathbf{b} \times \mathbf{c} \tag{1.59d}$$

$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c} \tag{1.59e}$$

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \cdot \mathbf{c})\mathbf{b} - (\mathbf{a} \cdot \mathbf{b})\mathbf{c} \tag{1.59f}$$

Finally, we can also find the following identities for the cross products between the basis vectors $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$.

$$\hat{\mathbf{x}} \times \hat{\mathbf{y}} = \hat{\mathbf{z}} \tag{1.60a}$$

$$\hat{\mathbf{y}} \times \hat{\mathbf{x}} = -\hat{\mathbf{z}} \tag{1.60b}$$

$$\hat{\mathbf{y}} \times \hat{\mathbf{z}} = \hat{\mathbf{x}} \tag{1.60c}$$

$$\hat{\mathbf{z}} \times \hat{\mathbf{y}} = -\hat{\mathbf{x}} \tag{1.60d}$$

$$\hat{\mathbf{z}} \times \hat{\mathbf{x}} = \hat{\mathbf{y}} \tag{1.60e}$$

$$\hat{\mathbf{x}} \times \hat{\mathbf{z}} = -\hat{\mathbf{y}} \tag{1.60f}$$

Note that

$$\begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \begin{bmatrix} b_z a_y - a_z b_y \\ b_x a_z - b_z a_x \\ b_y a_x - b_x a_y \end{bmatrix} \tag{1.61}$$

For a vector $\mathbf{a} = (a_x, a_y, a_z)^T$, let's define the **cross product matrix** $\mathbf{a}^\times$ as

$$\mathbf{a}^\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \tag{1.62}$$

Then, the cross product between $\mathbf{a}$ and $\mathbf{b}$ can also be expressed as a matrix multiplication:

$$\mathbf{a} \times \mathbf{b} = \mathbf{a}^\times \mathbf{b} \tag{1.63}$$

Also note that [22, 87]

$$\mathbf{a} \times \mathbf{b} = (\mathbf{b}^\times)^T \mathbf{a} \tag{1.64}$$

Given $\mathbf{a}^\times = [a_{ij}^\times]$, we can also recover $\mathbf{a}$ as

$$\mathbf{a} = \begin{bmatrix} a_{32}^\times \\ a_{13}^\times \\ a_{21}^\times \end{bmatrix} \tag{1.65}$$

Finally, we can also note that for a scalar $c \in \mathbb{R}$ and a vector $\mathbf{a} \in \mathbb{R}^3$,

$$c(\mathbf{a}^\times) = (c\mathbf{a})^\times \tag{1.66}$$

since

$$c(\mathbf{a}^\times) = c \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} = \begin{bmatrix} 0 & -ca_z & ca_y \\ ca_z & 0 & -ca_x \\ -ca_y & ca_x & 0 \end{bmatrix} = \begin{bmatrix} ca_x \\ ca_y \\ ca_z \end{bmatrix}^\times = (c\mathbf{a})^\times$$

Algorithms 6 and 7 below implement Eqs. (1.65) and (1.62).

Algorithm 6: `vec2skew`

Converts a vector to a skew-symmetric matrix representing a cross product.

Inputs:

- $\mathbf{a} = (a_1, a_2, a_3)^T \in \mathbb{R}^3$ - vector

Procedure:

$$\mathbf{a}^\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

   return $\mathbf{a}^\times$

Outputs:

- $\mathbf{a}^\times \in \mathbb{R}^{3\times3}$ - skew-symmetric matrix where $\mathbf{a} \times \mathbf{b} = \mathbf{a}^\times \mathbf{b}$

Test Cases:

- See Appendix A.4.

Algorithm 7: `skew2vec`

Converts a skew-symmetric matrix representing a cross product to a vector.

Inputs:

- $\mathbf{a}^\times \in \mathbb{R}^{3\times3}$ - skew-symmetric matrix where $\mathbf{a} \times \mathbf{b} = \mathbf{a}^\times \mathbf{b}$

Procedure:

$$\mathbf{a} = \begin{bmatrix} a_{32}^\times \\ a_{13}^\times \\ a_{21}^\times \end{bmatrix}$$

   return $\mathbf{a}$

Outputs:
- $\mathbf{a} \in \mathbb{R}^3$ - vector

Test Cases:
- See Appendix A.4.

Orthogonal vectors

The vectors $\mathbf{a}$ and $\mathbf{b}$ are orthogonal if

$$\mathbf{a} \cdot \mathbf{b} = 0 \qquad \text{or} \qquad \frac{\|\mathbf{a} \times \mathbf{b}\|}{\|\mathbf{a}\| \|\mathbf{b}\|} = 1$$

Parallel vectors

The vectors $\mathbf{a}$ and $\mathbf{b}$ are parallel if

$$\mathbf{a} \times \mathbf{b} = \mathbf{0} \qquad \text{or} \qquad \|\mathbf{a} \times \mathbf{b}\| = 0 \qquad \text{or} \qquad \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} = 1$$

Vector and scalar projections

The **vector projection** of $\mathbf{b}$ onto $\mathbf{a}$, denoted $\mathrm{proj}_{\mathbf{a}}\mathbf{b}$, is a vector in the direction of $\mathbf{a}$ whose magnitude is the component of $\mathbf{b}$ in the direction of $\mathbf{a}$. The vector projection is given by
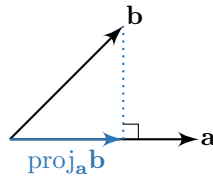
Figure 1.10: Vector projection.

$$\boxed{\mathrm{proj}_{\mathbf{a}}\mathbf{b} = \frac{\mathbf{a}\,(\mathbf{a} \cdot \mathbf{b})}{\|\mathbf{a}\|^2} = \left(\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|}\right) \frac{\mathbf{a}}{\|\mathbf{a}\|} = \left(\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|}\right) \hat{\mathbf{a}} = \text{vector projection of } \mathbf{b} \text{ onto } \mathbf{a}}$$

(1.67)

The **scalar projection** of $\mathbf{b}$ onto $\mathbf{a}$ is a scalar defined as the magnitude of the vector projection of $\mathbf{b}$ onto $\mathbf{a}$, or as the magnitude of $\mathbf{b}$ in the direction of $\mathbf{a}$.

$$\mathrm{comp}_{\mathbf{a}}\mathbf{b} = \left\|\left(\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|}\right) \hat{\mathbf{a}}\right\| = \|\mathrm{proj}_{\mathbf{a}}\mathbf{b}\| = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|}$$

$$\boxed{\mathrm{comp}_{\mathbf{a}}\mathbf{b} = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|} = \|\mathrm{proj}_{\mathbf{a}}\mathbf{b}\| = \text{scalar projection of } \mathbf{b} \text{ onto } \mathbf{a}}$$

(1.68)

Thus, we can also write the vector projection in terms of the scalar projection as

$$\boxed{\mathrm{proj}_{\mathbf{a}}\mathbf{b} = (\mathrm{comp}_{\mathbf{a}}\mathbf{b})\,\hat{\mathbf{a}} = \text{vector projection of } \mathbf{b} \text{ onto } \mathbf{a}}$$

(1.69)

Direction angles and direction cosines

The **direction angles** are the angles α, β, and γ, in the interval $[0, \pi]$, that $\mathbf{a}$ makes with the positive x-, y-, and z-axes, respectively. The **direction cosines** are the cosines of the direction angles and are given by
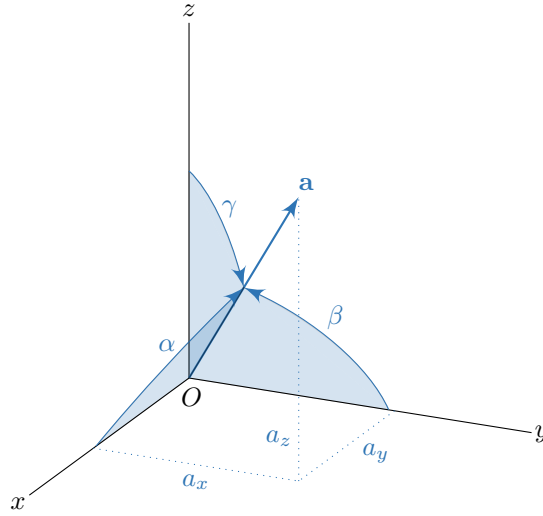
Figure 1.11: Direction angles.

$$\cos\alpha = \frac{\mathbf{a} \cdot \hat{\mathbf{x}}}{\|\mathbf{a}\|\,\|\hat{\mathbf{x}}\|} = \frac{a_x}{\|\mathbf{a}\|} = \frac{a_x}{a} = \frac{a_x}{\sqrt{a_x^2 + a_y^2 + a_z^2}} \qquad (1.70)$$

$$\cos\beta = \frac{\mathbf{a} \cdot \hat{\mathbf{y}}}{\|\mathbf{a}\|\,\|\hat{\mathbf{y}}\|} = \frac{a_y}{\|\mathbf{a}\|} = \frac{a_y}{a} = \frac{a_y}{\sqrt{a_x^2 + a_y^2 + a_z^2}} \qquad (1.71)$$

$$\cos\gamma = \frac{\mathbf{a} \cdot \hat{\mathbf{z}}}{\|\mathbf{a}\|\,\|\hat{\mathbf{z}}\|} = \frac{a_z}{\|\mathbf{a}\|} = \frac{a_z}{a} = \frac{a_z}{\sqrt{a_x^2 + a_y^2 + a_z^2}} \qquad (1.72)$$

Vector Triple Product

The vector triple product arises occasionally in kinematics and orbital mechanics. Given three column vectors $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^3$, the scalar triple product is defined as $\mathbf{a} \times (\mathbf{b} \times \mathbf{c})$. Using the triple product expansion[5], we can write the vector triple product in terms of dot products and elementary arithmetic as [99]

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \cdot \mathbf{c})\mathbf{b} - (\mathbf{a} \cdot \mathbf{b})\mathbf{c} \qquad (1.73)$$

---

[5]   Also known as "Lagrange's formula", "ACB-ABC rule", or "BAC-CAB rule" [99].

2

Physical Vectors

2.1 Physical Vectors and Matrices

Recall the column vector, $\mathbf{v} \in \mathbb{R}^n$, from Chapter 1. In the simplest sense, a column vector is just a list of numbers. For most of Chapter 1, we considered column vectors in coordinate systems (i.e. they were defined with respect to an ordered basis). However, there was no part of the vector itself that stored which coordinate system it was defined with respect to; it was simply a list of numbers. Thus, in the simplest sense, a list of numbers defines a column vector.

In dynamics, a **physical vector**[1] is an abstract quantity defined by some physical property; *not* by components or a coordinate system. More specifically, physical vectors describe physical quantities that have magnitude and direction in a three-dimensional space [39]. To describe our physical world, we attach coordinate systems to reference points. However, the physical world still exists without these coordinate systems, so physical vectors exist independent of coordinate systems as well, as shown in Fig. 2.1 below. This introduces some level of ambiguity, which allows

Figure 2.1: Physical vector.

us to write mathematical equations in a coordinate-free manner. The resulting equations are tantalizingly similar to matrix algebra, yet not quite the same. In most standard textbooks on dynamics and engineering, this distinction is either briefly mentioned or completely overlooked, as most examples and concepts can be taught without going into much depth. This becomes an issue when simulating complex physical systems using computers, which can only store column vectors. In this text, we emphasize this difference by devoting separate chapters to column and physical vectors. For an extremely clear, more in-depth discussion on this matter, see the introduction section of [86].

Note that in some sense, physical vectors *do* build upon column vectors; physical vectors can be **expressed** or **resolved** in a coordinate system. This will become more clear conceptually when we discuss coordinate vectors in Section 2.4.

---

[1] From this point on, we may refer to both column vectors and physical vectors simply as vectors. Using context clues, as well as the symbols used for the vectors, it should be clear which type of vector we are dealing with. In cases where we are using both physical vectors and column vectors, it will be made clear which is which.

> **Convention 13:** Notation for physical vectors.
>
> Physical vectors are written using lowercase, italic, boldface symbols. For example, a physical vector may be written as
> $$\boldsymbol{v}$$

While a physical vector has magnitude and direction, a **physical scalar** only has a magnitude. They are similar to mathematical scalars in the sense that they can be represented using a single number. However, there is some ambiguity, since physical magnitudes also depend on what **units** they are expressed in.

> **Convention 14:** Notation for physical scalars.
>
> Physical scalars are written using lowercase or uppercase, italic, non-boldface symbols. For example, a physical scalar may be written as
> $$a$$

The **magnitude** of a physical vector $\boldsymbol{v}$ is denoted as

$$\boxed{|\boldsymbol{v}| = v = \text{magnitude of } \boldsymbol{v}} \tag{2.1}$$

> **Convention 15:** Notation for the magnitude of a physical vector.
>
> Since the magnitude of a physical vector is a physical scalar, it is written using the italic, non-boldface version of the same symbol. For example, the magnitude of $\boldsymbol{v}$ is written as
> $$v$$

To describe the **direction** of a physical vector, we use unit vectors. A **unit vector** is a vector of magnitude 1. Therefore, if we multiply a scalar quantity by a unit vector, we get a vector whose magnitude is equal to the scalar and whose direction is parallel to the unit vector's. The unit vector in the direction of $\boldsymbol{v}$ is defined as

$$\boxed{\hat{\boldsymbol{v}} = \frac{\boldsymbol{v}}{|\boldsymbol{v}|} = \frac{\boldsymbol{v}}{v} = \text{direction of } \boldsymbol{v}} \tag{2.2}$$

> **Convention 16:** Notation for physical unit vectors.
>
> A physical unit vector in the direction of a physical vector is written using the same symbol but with a hat over it. For example, the unit vector in the direction of $\boldsymbol{v}$ is written as
> $$\hat{\boldsymbol{v}}$$

By rearranging Eq. (2.2), we can write a vector $\boldsymbol{v}$ in terms of its magnitude and direction.

$$\boxed{\boldsymbol{v} = v\hat{\boldsymbol{v}}} \tag{2.3}$$

We mentioned that in some way, physical vectors are related to column vectors. Similarly, we can introduce the concept of a **physical matrix**, that is related to a mathematical matrix. The simplest example of a physical matrix is an inertia tensor[2], I. An inertia tensor describing a three-dimensional body can be expressed in a three-dimensional coordinate system, resulting in a 3×3 mathematical matrix whose elements store the moments of inertia of the body with respect to the coordinate axes and planes.

> **Convention 17:** Notation for physical matrices.
>
> Physical matrices are written using uppercase, italic, boldface symbols. For example, a physical matrix may be written as
> $$A$$

2.2 Reference Frames

Just like a physical vector is an abstract notion of a column vector, a **reference frame** is an abstract notion of a coordinate system. It gives meaning to physical vectors in the sense that it defines what they are measured with respect to.

Reference frames are always fixed with respect to something in our universe. For example, we can fix a reference frame to the center of the Earth, such that as the Earth rotates, the reference frame rotates with it. Alternatively, we could fix a reference frame to the center of the Earth, but have it fixed with respect to the stars, such that the Earth rotates with respect to it. We can even fix a reference frame to a rocket, such that reference frame moves with the rocket (i.e. this reference frame would essentially be the point of view of an astronaut sitting at the top of the rocket). As another example, consider a car that is traveling down the freeway. From the reference frame of a hitchhiker standing on the side of the freeway, the occupants of the car are moving. From the frame of reference of the car, the occupants are not moving; they remain in their seats, not moving relative to the car.

Note that in the example above, we didn't include any coordinate systems; this is because reference frames are completely independent of coordinate systems. While reference frames are a vital concept from a theoretical viewpoint, in this book we will primarily interested in *coordinate* frames (see Section 2.3).

2.2.1 Inertial Reference Frames

An **inertial reference frame** is simply one that is not accelerating. If we lived in a stagnant universe that could be described simply using a Cartesian coordinate system, an inertial reference frame would be one that is stationary or moving with a constant velocity with respect to this coordinate system. Note that a reference frame that is rotating is also considered as accelerating; if we had a constant vector defined with respect to a rotating reference frame, the tip of that vector would trace a curved path with respect to a stationary reference frame, and if a vector changes direction it implies that it is not constant and therefore its derivative (i.e. velocity) is nonzero.

Unfortunately, gravity distorts space-time, and the universe is expanding, so it is technically impossible to define a "true" inertial reference frame. Nonetheless, we call some specific reference frames "inertial" as long as they are *mostly* inertial (which is also dependent on the context). For example, we often use an inertial frame that is fixed to the center of the Earth and whose orientation is stationary with respect to the distant stars. When considering dynamics of vehicles near Earth, such a frame can be considered to be inertial. However, the Earth itself is accelerating since it is travelling in orbits around the Sun. Therefore, any reference frame fixed to the Earth, regardless if its orientation is stationary with respect to the distant stars, cannot be truly inertial. As a result, if we are simulating the movement

---

[2] In mathematics, a tensor is generalization of a matrix to higher-dimensional vector spaces. In dynamics, an inertia tensor colloquially refers to a mathematical matrix storing the moments of inertia of a three-dimensional body. Note that the inertia tensor is also sometimes referred to simply as the inertia *matrix* [58, 94].

of the planetary bodies in our solar system, we will use a reference frame fixed to the center of the Sun and whose orientation is stationary with respect to the distant stars. But then we must recall that the Sun is orbiting the center of our galaxy, so any reference frame fixed to the Sun would also be non-inertial. We can continue this reasoning indefinitely, and still fail to find a truly inertial reference frame [42].

2.3 Coordinate Frames

Coordinate frames are the link between coordinate systems and reference frames. Coordinate systems can exist completely independently of reference frames, and reference frames can exist completely independently of coordinate systems. Coordinate frames are the special case where we attach a coordinate system *to* a reference frame in order to quantify vectors with respect to that reference frame. Essentially,

$$\boxed{(\text{coordinate system}) + (\text{reference frame}) = (\text{coordinate frame})}$$

> Note that in a coordinate frame, the coordinate system remains fixed to the reference frame; it can never move relative to the reference frame it describes.

> **Convention 18:** Referring to coordinate frames as frames.
>
> In many cases, we will refer to a coordinate frame simply as a "frame". This is because in the context of simulation, we deal almost exclusively with *coordinate* frames that have a coordinate system, as opposed to *reference* frames (which are a more abstract concept that do not inherently possess coordinate systems).

Consider a coordinate system with origin O and basis vectors $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$. We would call this coordinate system $Oxyz$ and refer to its ordered basis as $A = (\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$. The basis vectors ($\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$) in this case are *column* vectors; they are sets of numbers with no relation to the physical world. Conversely, in the case of coordinate frames, we begin with three **physical basis vectors**, $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, and $\hat{\boldsymbol{z}}$, that *do* point in a very specific direction in relation to our physical world. We say a coordinate frame, $\mathcal{A}$, is defined as a **physical ordered basis** consisting of physical basis vectors.

$$\mathcal{A} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}) = \text{physical ordered basis} = \text{coordinate frame}$$

> **Convention 19:** Coordinate frame / physical ordered basis notation for physical laws and generic kinematic relationships.
>
> For the purposes of defining physical laws or writing generic kinematic relationships, coordinate frames (a.k.a. a physical ordered basis) are denoted using an uppercase, calligraphic character, for example
> $$\mathcal{A}$$

Simply put, a coordinate frame is just a physical ordered basis. The physical basis vectors themselves are what "connect" the coordinate system to the physical world[3].

In aerospace, we frequently encounter (and usually deal with) coordinate frames that have very specific names and abbreviations. For example, the Earth-centered Earth-fixed frame is abbreviated to ECEF, the East, North, Up

---

[3] For example, the coordinate system for the Earth-centered inertial coordinate frame is defined such that its X axis, defined by the physical basis vector $\hat{\boldsymbol{X}}$, points towards the vernal equinox.

frame is abbreviated to ENU, etc. In these cases, we will typically refer to the frame in text using their abbreviations. In the context of mathematical equations, the coordinate frames are denoted using the abbreviation in all *lowercase*.

> **Convention 20:** Coordinate frame notation for specific coordinate frames.
>
> When dealing with physical or mathematical quantities relating to a specific coordinate frame, the coordinate frame is denoted using its abbreviation in all lowercase. For example, the position of a vehicle, r, expressed[a] in the Earth-Centered Earth-fixed coordinate frame (abbreviated to ECEF) would be
>
> $$[r]_{\text{ecef}}$$
>
> _____
> [a] See Section 2.4 for more on coordinate vectors.

2.3.1 Inertial Coordinate Frames

An **inertial coordinate frame** is simply the combination of an inertial reference frame (see Section 2.2.1) with a coordinate system fixed to said reference frame. Most physical laws and equations are defined specifically for inertial coordinate frames, but are also general to *any* inertial coordinate frame (i.e. the equations and laws hold for any inertial coordinate frame). Therefore, we must define a simple way to denote a generic inertial coordinate frame.

> **Convention 21:** Generic inertial coordinate frame notation.
>
> A generic inertial coordinate frame is denoted using an uppercase, calligraphic I.
>
> $$\mathcal{I}$$

2.4 Coordinate Vectors

Just how coordinate frames are the link between coordinate systems and reference frames, **coordinate vectors** are the link between column vectors and physical vectors. Consider a physical vector, v. Now, consider a coordinate frame $\mathcal{A}$ with the physical ordered basis $(\hat{x}, \hat{y}, \hat{z})$ defining the coordinate system $Oxyz$. Similar to how column vectors could be written in terms of the basis vectors of a coordinate *system* in Eq. (1.8), we can write physical vectors in terms of the basis vectors of a coordinate *frame*.

$$\boxed{\boldsymbol{v} = v_x\hat{\boldsymbol{x}} + v_y\hat{\boldsymbol{y}} + v_z\hat{\boldsymbol{z}}} \tag{2.4}$$

v_x, v_y, and v_z are the **coordinates** or **components** of v.

> As an aside, we can define the magnitude, v, and direction, $\hat{v}$, in terms of its components.
>
> $$\boxed{v = \sqrt{v_x^2 + v_y^2 + v_z^2}} \tag{2.5}$$
>
> $$\boxed{\hat{\boldsymbol{v}} = \left(\frac{v_x}{v}\right)\hat{\boldsymbol{x}} + \left(\frac{v_y}{v}\right)\hat{\boldsymbol{y}} + \left(\frac{v_z}{v}\right)\hat{\boldsymbol{z}}} \tag{2.6}$$

Let's rewrite Eq. (2.4) using matrix algebra.

$$v = \underbrace{\begin{bmatrix} \hat{x} & \hat{y} & \hat{z} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}}_{[v]_\mathcal{A}} \tag{2.7}$$

We define the **frame matrix** as

$$\boxed{A = \begin{bmatrix} \hat{x} & \hat{y} & \hat{z} \end{bmatrix}} \tag{2.8}$$

and the **coordinate vector** of v when expressed in coordinate frame $\mathcal{A}$ as [21], [51, pp. 163–172]

$$\boxed{[v]_\mathcal{A} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}} \tag{2.9}$$

Convention 22: Coordinate vector notation.

Case #1: Generic Coordinate Frames
A physical vector expressed in a generic coordinate frame should be placed in brackets and subscripted (outside the brackets) with the symbol representing the coordinate frame. For example, a physical vector, v, expressed in coordinate frame $\mathcal{A}$ should be denoted

$$[v]_\mathcal{A}$$

Case #2: Specific Coordinate Frames
If we are dealing with a specific coordinate frame, such as the ECEF frame, we use the lowercase of the frame abbreviation as the subscript. For example, a physical vector, v, expressed in the ECEF coordinate frame should be denoted

$$[v]_\text{ecef}$$

The reason we use lowercase is because the uppercase lettering can make the subscript look very large when subscripting lowercase symbols.

Additional Note: Descriptive Subscripts
If the vector has a descriptive subscript, the descriptive subscript remains inside the brackets, while the frame subscript is outside the brackets. For example, if a vector has the descriptive subscript "ex", the corresponding coordinate vector expressed in the coordinate frame $\mathcal{A}$ is

$$[v_\text{ex}]_\mathcal{A}$$

From Eq. (2.7), we can write v in terms of A and $[v]_\mathcal{A}$ as

$$\boxed{v = A[v]_\mathcal{A}} \tag{2.10}$$

Note that a frame matrix can also be expressed in a coordinate frame. For example, if we have another coordinate frame $\mathcal{B} = (\hat{a}, \hat{b}, \hat{c})$,

$$\boxed{[A]_\mathcal{B} = \left([\hat{x}]_\mathcal{B} \quad [\hat{y}]_\mathcal{B} \quad [\hat{z}]_\mathcal{B} \right)} \tag{2.11}$$

2.5   **Frame Transformation via Passive Rotation**

Consider the physical vector v and the coordinate frames $\mathcal{A} = (\hat{x}, \hat{y}, \hat{z})$ and $\mathcal{B} = (\hat{a}, \hat{b}, \hat{v})$. Using Eq. (2.10), we express v in either coordinate frame.

$$v = A[v]_{\mathcal{A}} = B[v]_{\mathcal{B}}$$

In the equation above, A and B are the frame matrices given by

$$A = \begin{bmatrix} \hat{x} & \hat{y} & \hat{z} \end{bmatrix}, \qquad B = \begin{bmatrix} \hat{a} & \hat{b} & \hat{c} \end{bmatrix}$$

while $[v]_{\mathcal{A}}$ and $[v]_{\mathcal{B}}$ are the coordinate vectors of v expressed in bases $\mathcal{A}$ and $\mathcal{B}$, respectively.

$$[v]_{\mathcal{A}} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}, \qquad [v]_{\mathcal{B}} = \begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix}$$

Consider the case where we have knowledge of the basis vectors of $\mathcal{A}$ expressed in coordinate frame $\mathcal{B}$. Then we have the coordinate matrix

$$[A]_{\mathcal{B}} = \left([\hat{x}]_{\mathcal{B}} \quad [\hat{y}]_{\mathcal{B}} \quad [\hat{z}]_{\mathcal{B}} \right)$$

We know from Eq. (1.20) that

$$[v]_{\mathcal{B}} = [A]_{\mathcal{B}}[v]_{\mathcal{A}}$$

Furthermore, we have $[v]_{\mathcal{A}}, [v]_{\mathcal{B}} \in \mathbb{R}^3$ and $[A]_{\mathcal{B}} \in \mathbb{R}^{3\times3}$, so $[A]_{\mathcal{B}}$ represents a passive rotation matrix (see Section **??**). Thus, we write

$$[v]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}[v]_{\mathcal{A}} \tag{2.12}$$

where $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$ is the passive rotation matrix from coordinate frame $\mathcal{A}$ to coordinate frame $\mathcal{B}$.

Convention 23: Notation for a passive rotation matrix between physical bases.

Consider the physical vector v and the coordinate frames $\mathcal{A} = (\hat{x}, \hat{y}, \hat{z})$ and $\mathcal{B} = (\hat{a}, \hat{b}, \hat{v})$. The passive rotation matrix from coordinate frame $\mathcal{A}$ to coordinate frame $\mathcal{B}$ is written as

$$\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$$

and satisfies

$$[v]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}[v]_{\mathcal{A}}$$

Algorithm 8: `matrotate`
Passive rotation of a vector by a rotation matrix.

Inputs:
- $[\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from coordinate frame $\mathcal{A}$ to coordinate frame $\mathcal{B}$
- $[r]_{\mathcal{A}} \in \mathbb{R}^3$ - vector expressed in coordinate frame $\mathcal{A}$

Procedure:

$$[r]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}[r]_{\mathcal{A}}$$
return $[r]_{\mathcal{B}}$

Outputs:
- $[r]_{\mathcal{B}} \in \mathbb{R}^3$ - vector expressed in coordinate frame $\mathcal{B}$

Test Cases:

- See Appendix A.1.1.

If we wanted to solve for $[v]_\mathcal{B}$ given $[v]_\mathcal{A}$, we could left-multiply both sides by $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T$ (since the inverse of a rotation matrix is its transpose).

$$[v]_\mathcal{A} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T [v]_\mathcal{B}$$

However, note that

$$[v]_\mathcal{A} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}} [v]_\mathcal{B}$$

Thus, the rotation matrix for the inverse transformation is

$$\boxed{\mathbf{R}_{\mathcal{B}\to\mathcal{A}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T} \tag{2.13}$$

2.5.1   Chained Rotations

Let $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$ be the passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$, and let $\mathbf{R}_{\mathcal{B}\to\mathcal{C}}$ be the passive rotation matrix from frame $\mathcal{B}$ to frame $\mathcal{C}$. Let's start with a physical vector, r, expressed in frame $\mathcal{A}$, and find the coordinates of that vector expressed in frame $\mathcal{B}$.

$$[r]_\mathcal{B} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}} [r]_\mathcal{A} \tag{2.14}$$

Now that we know $[r]_\mathcal{B}$, we can find $[r]_\mathcal{C}$ using $\mathbf{R}_{\mathcal{B}\to\mathcal{C}}$.

$$[r]_\mathcal{C} = \mathbf{R}_{\mathcal{B}\to\mathcal{C}} [r]_\mathcal{B} \tag{2.15}$$

Our goal is to perform this **chained rotation** in a single step as

$$[r]_\mathcal{C} = \mathbf{R}_{\mathcal{A}\to\mathcal{C}} [r]_\mathcal{A}$$

Substituting Eq. (2.14) into Eq. (2.15),

$$[r]_\mathcal{C} = \mathbf{R}_{\mathcal{B}\to\mathcal{C}} \left(\mathbf{R}_{\mathcal{A}\to\mathcal{B}} [r]_\mathcal{A} \right)$$
$$= \underbrace{\left(\mathbf{R}_{\mathcal{B}\to\mathcal{C}} \mathbf{R}_{\mathcal{A}\to\mathcal{B}} \right)}_{\mathbf{R}_{\mathcal{A}\to\mathcal{C}}} [r]_\mathcal{A}$$

Thus, we have the following rule for chaining together rotation matrices:

$$\boxed{\mathbf{R}_{\mathcal{A}\to\mathcal{C}} = \mathbf{R}_{\mathcal{B}\to\mathcal{C}} \mathbf{R}_{\mathcal{A}\to\mathcal{B}}} \tag{2.16}$$

This chaining can be extended indefinitely. For example,

$$\mathbf{R}_{\mathcal{A}\to\mathcal{E}} = \mathbf{R}_{D\to E} \mathbf{R}_{\mathcal{C}\to\mathcal{D}} \mathbf{R}_{\mathcal{B}\to\mathcal{C}} \mathbf{R}_{\mathcal{A}\to\mathcal{B}}$$

Algorithm 9: `matchain`
Chaining passive rotations represented by rotation matrices.

Inputs:
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from coordinate frame $\mathcal{A}$ to coordinate frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{B}\to\mathcal{C}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from coordinate frame $\mathcal{B}$ to coordinate frame $\mathcal{C}$

Procedure:

$$\mathbf{R}_{\mathcal{A}\to\mathcal{C}} = \mathbf{R}_{\mathcal{B}\to\mathcal{C}}\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$$
return $\mathbf{R}_{\mathcal{A}\to\mathcal{C}}$

Outputs:
- $\mathbf{R}_{\mathcal{A}\to\mathcal{C}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from coordinate frame $\mathcal{A}$ to coordinate frame $\mathcal{C}$

Test Cases:
- See Appendix A.1.1.

2.5.2 Constructing Rotation Matrices from Basis Vectors

Earlier, we skimmed over the fact that the passive rotation matrix from coordinate frame A to coordinate frame B is simply

$$\boxed{\mathbf{R}_{\mathcal{A}\to\mathcal{B}} = [A]_{\mathcal{B}}} \tag{2.17}$$

where

$$[A]_{\mathcal{B}} = \left([\hat{\boldsymbol{x}}]_{\mathcal{B}} \quad [\hat{\boldsymbol{y}}]_{\mathcal{B}} \quad [\hat{\boldsymbol{z}}]_{\mathcal{B}}\right)$$

is the coordinate matrix of coordinate frame A expressed in coordinate frame B. Thus, if we know the unit vectors of one frame expressed in the other frame, then we can form a passive rotation matrix between the two frames.

$$\boxed{\mathbf{R}_{\mathcal{A}\to\mathcal{B}} = \left([\hat{\boldsymbol{x}}]_{\mathcal{B}} \quad [\hat{\boldsymbol{y}}]_{\mathcal{B}} \quad [\hat{\boldsymbol{z}}]_{\mathcal{B}}\right) \qquad ([\hat{\boldsymbol{x}}]_{\mathcal{B}}, [\hat{\boldsymbol{y}}]_{\mathcal{B}}, \text{ and } [\hat{\boldsymbol{z}}]_{\mathcal{B}} \text{ are the basis vectors of } A \text{ expressed in } B)} \tag{2.18}$$

Conversely, we also know that

$$\mathbf{R}_{\mathcal{B}\to\mathcal{A}} = \left([\hat{\boldsymbol{a}}]_{\mathcal{A}} \quad [\hat{\boldsymbol{b}}]_{\mathcal{A}} \quad [\hat{\boldsymbol{c}}]_{\mathcal{A}}\right)$$

where $[\hat{\boldsymbol{a}}]_{\mathcal{A}}$, $[\hat{\boldsymbol{b}}]_{\mathcal{A}}$, and $[\hat{\boldsymbol{c}}]_{\mathcal{A}}$ are the basis vectors of frame B expressed in frame A. Since $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}^{T}$,

$$\mathbf{R}_{\mathcal{A}\to\mathcal{B}} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}^{T} = \left([\hat{\boldsymbol{a}}]_{\mathcal{A}} \quad [\hat{\boldsymbol{b}}]_{\mathcal{A}} \quad [\hat{\boldsymbol{c}}]_{\mathcal{A}}\right)^{T}$$

$$\boxed{\mathbf{R}_{\mathcal{A}\to\mathcal{B}} = \begin{pmatrix} [\hat{\boldsymbol{a}}]_{\mathcal{A}}^{T} \\ [\hat{\boldsymbol{b}}]_{\mathcal{A}}^{T} \\ [\hat{\boldsymbol{c}}]_{\mathcal{A}}^{T} \end{pmatrix} \qquad ([\hat{\boldsymbol{a}}]_{\mathcal{A}}, [\hat{\boldsymbol{b}}]_{\mathcal{A}}, \text{ and } [\hat{\boldsymbol{c}}]_{\mathcal{A}} \text{ are the basis vectors of } B \text{ expressed in } A)} \tag{2.19}$$

2.6 The Algebra of Physical Vectors

In Section 2.1, we described how we often write physics/dynamics equations in terms of physical vectors, but in practice (especially in simulation) we need to write them in terms of coordinate vectors that we can actually perform arithmetic with. Let's consider the simplest possible equation below, where we add two physical vectors.

$$\boldsymbol{c} = \boldsymbol{a} + \boldsymbol{b}$$

Now, let's introduce a coordinate frame, F. We can write this equation in terms of the coordinate vectors expressed in frame F by using the frame matrix, $\boldsymbol{F}$, corresponding to frame F (see Section 2.4).

$$\boldsymbol{F}[\boldsymbol{c}]_F = \boldsymbol{F}[\boldsymbol{a}]_F + \boldsymbol{F}[\boldsymbol{b}]_F \tag{2.20}$$

Recall that a frame matrix is internally composed of three physical basis vectors. In this case,

$$\boldsymbol{F} = \begin{bmatrix} \hat{\boldsymbol{x}} & \hat{\boldsymbol{y}} & \hat{\boldsymbol{z}} \end{bmatrix}$$

This is analogous to the case of writing a 3×3 mathematical matrix $\mathbf{A} \in \mathbb{R}^{3 \times 3}$ in terms of its three column vectors, $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3 \in \mathbb{R}^3$.

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \mathbf{a}_3 \end{bmatrix}$$

Since we can perform any matrix algebra in terms of partitioned matrices[4] (in this case, we are partitioning the matrix into its columns), we can just interpret a frame matrix as a partitioned matrix (this time partitioned into three physical vectors). Although the blocks of the frame matrix are abstract quantities, we can just treat them as any other vector. Thus, continuing our example from Eq. (2.20),

$$\boldsymbol{F}[\boldsymbol{c}]_F = \boldsymbol{F} \left([\boldsymbol{a}]_F + [\boldsymbol{b}]_F \right)$$

This implies that

$$[\boldsymbol{c}]_F = [\boldsymbol{a}]_F + [\boldsymbol{b}]_F$$

Recall the equation we started with at the beginning of this section. We have essentially showed that the coordinate vector form of an equation is largely analogous to the physical vector form.

$$\boldsymbol{c} = \boldsymbol{a} + \boldsymbol{b} \quad \Longleftrightarrow \quad [\boldsymbol{c}]_F = [\boldsymbol{a}]_F + [\boldsymbol{b}]_F$$

> **The algebra of physical vectors and matrices is the same as the algebra of mathematical vectors and matrices.**
>
> Take this statement with a grain of salt; we did no rigorous proofs here and merely presented the simplest example. While we could easily go more in-depth, this is enough to demonstrate how the algebra of physical vectors and matrices is identical to that of mathematical vectors and matrices for our purposes.

2.6.1   Substituting Coordinate and Physical Vectors

Consider an arbitrary physical vector $\boldsymbol{s}$ and the coordinate frames $\mathcal{A}$ and $\mathcal{B}$ (with frame matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively). Recall that we can write $\boldsymbol{s}$ in terms of its coordinates vectors expressed in either of these frames as

$$\boldsymbol{s} = \boldsymbol{A}[\boldsymbol{s}]_\mathcal{A} = \boldsymbol{B}[\boldsymbol{s}]_\mathcal{B} \tag{2.21}$$

Also recall that we can write the coordinate vectors in terms of one another using the passive rotation matrix between the two frames.

$$[\boldsymbol{s}]_\mathcal{B} = \mathbf{R}_{\mathcal{A} \to \mathcal{B}}[\boldsymbol{s}]_\mathcal{A}$$

Substituting this into the first equation, we have

$$\boldsymbol{s} = \boldsymbol{A}[\boldsymbol{s}]_\mathcal{A} = \boldsymbol{B}\mathbf{R}_{\mathcal{A} \to \mathcal{B}}[\boldsymbol{s}]_\mathcal{A}$$

Thus, we can always make the following substitutions:

$$\boxed{\begin{aligned} \boldsymbol{s} &= \boldsymbol{A}[\boldsymbol{s}]_\mathcal{A} \\ \boldsymbol{s} &= \boldsymbol{B}\mathbf{R}_{\mathcal{A} \to \mathcal{B}}[\boldsymbol{s}]_\mathcal{A} \end{aligned}} \tag{2.22}$$

By comparing the two lines of Eq. (2.22), we can also see that

$$\boxed{\boldsymbol{A} = \boldsymbol{B}\mathbf{R}_{\mathcal{A} \to \mathcal{B}}} \tag{2.23}$$

While we easily arrived at Eq. (2.22) above, we extend the general idea provided by this equation below, but provide no proof. Instead, we will provide an example that should illustrate the concept, but nonetheless we concede that this statement can appear quite "hand-wavy".

---

[4]   See https://en.wikipedia.org/wiki/Block_matrix.

> In Eq. (2.22), if $[s]_\mathcal{A}$ is a coordinate vector expressed in frame $\mathcal{A}$ that is the result of some vector operations on coordinate vectors, then s can be defined using that same sequence of operations but on the *physical* vectors that the coordinate vectors describe.

The statement above is best illustrated through an example. Consider three coordinate frames, $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, and three coordinate vectors $[x]_\mathcal{A}$, $[y]_\mathcal{B}$, $[z]_\mathcal{C}$. Say we have the expression

$$B\left[(\mathbf{R}_{\mathcal{A}\to\mathcal{B}}[x]_\mathcal{A}) \times ([y]_\mathcal{B}) \times (\mathbf{R}_{C\to B}[z]_\mathcal{C})\right]$$

Since $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}[x]_\mathcal{A}$, $[y]_\mathcal{B}$, and $\mathbf{R}_{C\to B}[z]_\mathcal{C}$ are all coordinate vectors expressed in frame $\mathcal{B}$, the result of the triple cross product is also a coordinate vector expressed in frame $\mathcal{B}$. Furthermore, since we are multiplying by the frame matrix B, Eq. (2.22) implies

$$x \times y \times z = B\left[(\mathbf{R}_{\mathcal{A}\to\mathcal{B}}[x]_\mathcal{A}) \times ([y]_\mathcal{B}) \times (\mathbf{R}_{C\to B}[z]_\mathcal{C})\right]$$

> In Section **??**, we will use these substitutions to convert an equation involving coordinate vectors to an equation expressed using only physical vectors. However, in most cases, we will be taking the opposite route; we will have some physical law written in terms of physical vectors that we will want to express in terms of coordinate vectors in order to be able to implement in a simulation.

2.7   Physical Vector Operations

In the previous section, we discussed how the algebra of physical vectors mimics that of mathematical vectors. Additionally, since physical vectors are tied to a three-dimensional world, we can define some of the same operations for physical vectors as we did for mathematical vectors in Section **??**.

Dot (scalar) product

The **dot (scalar) product** between two physical vectors, a and b, is defined as

$$\boxed{a \cdot b = |a|\,|b|\cos\theta = ab\cos\theta} \tag{2.24}$$

where θ is the angle between the two vectors. Note that regardless of which coordinate frames the vectors are defined in, there *always* exists some angle θ between them.

> It follows that the dot product is also frame-independent; the magnitudes of a and b are already independent of any coordinate frame since they describe the *magnitude* of the vectors, not their *direction*.

The dot product also exhibits the following properties (where a, b, and c are physical vectors and k is an arbitrary constant).

$$a \cdot a = |a|^2 = a^2 \tag{2.25a}$$

$$a \cdot (b + c) = a \cdot b + a \cdot c \tag{2.25b}$$

$$0 \cdot a = 0 \tag{2.25c}$$

$$a \cdot b = b \cdot a \tag{2.25d}$$

$$(ka) \cdot b = k(a \cdot b) = a \cdot (kb) \tag{2.25e}$$

Finally, we can also find the following identities for the dot products between the basis vectors $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, and $\hat{\boldsymbol{z}}$.

$$\boxed{\hat{\boldsymbol{x}} \cdot \hat{\boldsymbol{y}} = \hat{\boldsymbol{y}} \cdot \hat{\boldsymbol{z}} = \hat{\boldsymbol{z}} \cdot \hat{\boldsymbol{x}} = 0}$$ (2.26a)

$$\boxed{\hat{\boldsymbol{x}} \cdot \hat{\boldsymbol{x}} = \hat{\boldsymbol{y}} \cdot \hat{\boldsymbol{y}} = \hat{\boldsymbol{z}} \cdot \hat{\boldsymbol{z}} = 1}$$ (2.26b)

Cross (vector) product

The **cross (vector) product** of $\boldsymbol{a}$ and $\boldsymbol{b}$ is a vector multiplication that produces a third vector that is normal/orthogonal/perpendicular to both $\boldsymbol{a}$ and $\boldsymbol{b}$.

$$\boxed{|\boldsymbol{a} \times \boldsymbol{b}| = |\boldsymbol{a}|\,|\boldsymbol{b}| \sin\theta = ab \sin\theta}$$ (2.27)

The cross product also exhibits the following properties (where $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ are vectors and k is an arbitrary constant).

$$\boldsymbol{a} \times \boldsymbol{b} = -\boldsymbol{b} \times \boldsymbol{a}$$ (2.28a)

$$(k\boldsymbol{a}) \times \boldsymbol{b} = k(\boldsymbol{a} \times \boldsymbol{b}) = \boldsymbol{a} \times (k\boldsymbol{b})$$ (2.28b)

$$\boldsymbol{a} \times (\boldsymbol{b} + \boldsymbol{c}) = \boldsymbol{a} \times \boldsymbol{b} + \boldsymbol{a} \times \boldsymbol{c}$$ (2.28c)

$$(\boldsymbol{a} + \boldsymbol{b}) \times \boldsymbol{c} = \boldsymbol{a} \times \boldsymbol{c} + \boldsymbol{b} \times \boldsymbol{c}$$ (2.28d)

$$\boldsymbol{a} \cdot (\boldsymbol{b} \times \boldsymbol{c}) = (\boldsymbol{a} \times \boldsymbol{b}) \cdot \boldsymbol{c}$$ (2.28e)

$$\boldsymbol{a} \times (\boldsymbol{b} \times \boldsymbol{c}) = (\boldsymbol{a} \cdot \boldsymbol{c})\boldsymbol{b} - (\boldsymbol{a} \cdot \boldsymbol{b})\boldsymbol{c}$$ (2.28f)

Finally, we can also find the following identities for the cross products between the basis vectors $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, and $\hat{\boldsymbol{z}}$.

$$\hat{\boldsymbol{x}} \times \hat{\boldsymbol{y}} = \hat{\boldsymbol{z}}$$ (2.29a)

$$\hat{\boldsymbol{y}} \times \hat{\boldsymbol{x}} = -\hat{\boldsymbol{z}}$$ (2.29b)

$$\hat{\boldsymbol{y}} \times \hat{\boldsymbol{z}} = \hat{\boldsymbol{x}}$$ (2.29c)

$$\hat{\boldsymbol{z}} \times \hat{\boldsymbol{y}} = -\hat{\boldsymbol{x}}$$ (2.29d)

$$\hat{\boldsymbol{z}} \times \hat{\boldsymbol{x}} = \hat{\boldsymbol{y}}$$ (2.29e)

$$\hat{\boldsymbol{x}} \times \hat{\boldsymbol{z}} = -\hat{\boldsymbol{y}}$$ (2.29f)

Orthogonal vectors

The vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ are orthogonal if

$$\boldsymbol{a} \cdot \boldsymbol{b} = 0 \qquad \text{or} \qquad \frac{|\boldsymbol{a} \times \boldsymbol{b}|}{|\boldsymbol{a}|\,|\boldsymbol{b}|} = 1$$

Parallel vectors

The vectors a and b are parallel if

$$a \times b = 0 \qquad \text{or} \quad |a \times b| = 0 \qquad \text{or} \qquad \frac{a \cdot b}{|a|\,|b|} = 1$$

Vector Triple Product

Like in the column vector case in Section **??**, we can rewrite the vector triple product for physical vectors in terms of dot products and elementary arithmetic.

$$\boxed{a \times (b \times c) = (a \cdot c)b - (a \cdot b)c} \tag{2.30}$$

3

Translational Kinematics

TODO: velocity always tangent to trajectory TODO: section on A and B relative motion (i.e. valid if A and B both rotating) vs. I and R relative motion (inertial and rotating) TODO: chained coordinate frames TODO: algorithms

Translational kinematics are fundamentally linked to the time derivatives of vectors. When accelerating (rotating and/or translating) reference frames are involved, taking the time derivative of vectors can become quite involved. In developing translational kinematics, most resources present no distinction between mathematical (column) vectors and physical vectors, and derive fundamental kinematic relationships using physical vectors alone. Furthermore, many derivations for the differentiation of vectors build upon some assumed prior knowledge of translational kinematics (specifically regarding angular velocity and the relationship between position and velocity), even though the relationship between translational kinematics and the differentiation of vectors is in reality reversed (fundamentally, translational kinematics build upon the differentiation of vectors).

In this chapter, we begin by defining the derivatives of mathematical (column) vectors in Section **??**, independent of any notion of a coordinate frame. In Section **??**, we then find the time derivatives of n-dimensional coordinate vectors in a rotating coordinate frame, completely independent of 3-dimensional physical vectors. In Section **??**, we begin by restricting the time derivatives of coordinate vectors to the 3-dimensional case, after which we use the relationship between physical and coordinate vectors (developed in Chapter 2) to obtain the time derivative of physical vectors. The angular velocity arises naturally from the first three sections of this chapter, and we devote Sections 3.4 and 3.5 to further developing the angular velocity and angular acceleration vectors. In Section 3.6, we formalize the *Golden Rule* of kinematics, which represents a transition in how we approach developing kinematic relationships. We begin the chapter by deriving relationships in terms of mathematical quantities and then abstracting them to physical relationships, but using the *Golden Rule* we begin taking the opposite approach (deriving physical relationships and then expressing them using mathematical quantities in certain coordinate frames).

Finally, we make a small note on the notation we will use in this chapter. In Chapters 1 and 2, we use the letter "v" (stylized as $\mathbf{v}$ and v) to denote vectors. In this chapter, we will find that this letter is reserved specifically for denoting velocities, so we instead switch to using the letter "s" to denote arbitrary vector quantities.

3.1 Time Derivatives of Mathematical Vectors

Consider a mathematical vector, $\mathbf{s} \in \mathbb{R}^n$.

$$\mathbf{s} = \begin{bmatrix} s_1 \\ \vdots \\ s_n \end{bmatrix}$$

Let's say that $\mathbf{s}$ is a function of a scalar independent variable, $t \in \mathbb{R}$.

$$\mathbf{s} = \begin{bmatrix} s_1(t) \\ \vdots \\ s_n(t) \end{bmatrix}$$

Then the derivative of $\mathbf{s}$ with respect to t is

$$\boxed{\frac{d\mathbf{s}}{dt} = \begin{bmatrix} \dfrac{ds_1}{dt} \\ \vdots \\ \dfrac{ds_n}{dt} \end{bmatrix} \in \mathbb{R}^n} \tag{3.1}$$

The time derivative of mathematical vectors can also be written using Newton's notation for time derivatives.

$$\dot{\mathbf{s}} \equiv \frac{d\mathbf{s}}{dt}$$

$$\ddot{\mathbf{s}} \equiv \frac{d^2\mathbf{s}}{dt^2}$$

3.2   Time Derivatives of Coordinate Vectors

Consider the coordinate vectors $[\mathbf{s}]_{\mathcal{X}} \in \mathbb{R}^n$ and $[\mathbf{s}]_{\mathcal{Y}} \in \mathbb{R}^n$, which represent the same vector expressed in two separate coordinate systems with the same origin, $Ox_1...x_n$ and $Oy_1...y_n$, respectively. These coordinate systems are defined by the ordered bases $\mathcal{X} = (\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_n)$ and $\mathcal{Y} = (\hat{\mathbf{y}}_1, ..., \hat{\mathbf{y}}_n)$, respectively.

$$[\mathbf{s}]_{\mathcal{X}} = \begin{bmatrix} s_{x_1} \\ \vdots \\ s_{x_n} \end{bmatrix}, \qquad [\mathbf{s}]_{\mathcal{Y}} = \begin{bmatrix} s_{y_1} \\ \vdots \\ s_{y_n} \end{bmatrix}$$

Allowing them to vary as functions of time, $t \in \mathbb{R}$,

$$[\mathbf{s}]_{\mathcal{X}}(t) = \begin{bmatrix} s_{x_1}(t) \\ \vdots \\ s_{x_n}(t) \end{bmatrix}, \qquad [\mathbf{s}]_{\mathcal{Y}}(t) = \begin{bmatrix} s_{y_1}(t) \\ \vdots \\ s_{y_n}(t) \end{bmatrix}$$

If we know both $[\mathbf{s}]_{\mathcal{X}}$ and $[\mathbf{s}]_{\mathcal{Y}}$, then their time derivatives are simply

$$\boxed{\frac{d[\mathbf{s}]_{\mathcal{X}}}{dt} = \begin{bmatrix} \dfrac{ds_{x_1}}{dt} \\ \vdots \\ \dfrac{ds_{x_n}}{dt} \end{bmatrix} \in \mathbb{R}^n, \qquad \frac{d[\mathbf{s}]_{\mathcal{Y}}}{dt} = \begin{bmatrix} \dfrac{ds_{y_1}}{dt} \\ \vdots \\ \dfrac{ds_{y_n}}{dt} \end{bmatrix} \in \mathbb{R}^n} \tag{3.2}$$

Now, let's consider the case where we know $[\mathbf{s}]_{\mathcal{X}}(t)$ and $[\mathbf{X}]_{\mathcal{Y}}(t)$, and want to find the derivative of $[\mathbf{s}]_{\mathcal{Y}}(t)$ with respect to t. Note that

$$[\mathbf{X}]_{\mathcal{Y}}(t) = \begin{pmatrix} [\hat{\mathbf{x}}_1]_{\mathcal{Y}} & \cdots & [\hat{\mathbf{x}}_n]_{\mathcal{Y}} \end{pmatrix}$$

is the basis matrix of $\mathcal{X}$ expressed in the basis $\mathcal{Y}$. From Section 1.4, we know we can write $[\mathbf{s}]_{\mathcal{Y}}$ in terms of $[\mathbf{s}]_{\mathcal{X}}$ as

$$[\mathbf{s}]_{\mathcal{Y}} = [\mathbf{X}]_{\mathcal{Y}}[\mathbf{s}]_{\mathcal{X}}$$

Differentiating both sides with respect to t,

$$\boxed{\frac{d[\mathbf{s}]_{\mathcal{Y}}}{dt} = \left(\frac{d[\mathbf{X}]_{\mathcal{Y}}}{dt}\right)[\mathbf{s}]_{\mathcal{X}} + [\mathbf{X}]_{\mathcal{Y}}\left(\frac{d[\mathbf{s}]_{\mathcal{X}}}{dt}\right)} \tag{3.3}$$

3.2.1 Orthogonal Bases

If $\mathcal{X}$ and $\mathcal{Y}$ are both *orthogonal* bases, then we know that

$$[\mathbf{X}]_{\mathcal{Y}}^{T}[\mathbf{X}]_{\mathcal{Y}} = \mathbf{I}_{n \times n}$$

Differentiating both sides with respect to t,

$$\left(\frac{d[\mathbf{X}]_{\mathcal{Y}}}{dt}\right)[\mathbf{X}]_{\mathcal{Y}}^{T} + [\mathbf{X}]_{\mathcal{Y}}\left(\frac{d[\mathbf{X}]_{\mathcal{Y}}^{T}}{dt}\right) = \mathbf{0}_{n \times n}$$

Noting that

$$\frac{d\mathbf{M}^{T}}{dt} = \left(\frac{d\mathbf{M}}{dt}\right)^{T}$$

for an arbitrary matrix $\mathbf{M}$, we have

$$\left(\frac{d[\mathbf{X}]_{\mathcal{Y}}}{dt}\right)[\mathbf{X}]_{\mathcal{Y}}^{T} + [\mathbf{X}]_{\mathcal{Y}}\left(\frac{d[\mathbf{X}]_{\mathcal{Y}}}{dt}\right)^{T} = \mathbf{0}_{n \times n} \tag{3.4}$$

Now, let's define

$$\boxed{[\mathbf{W}]_{\mathcal{Y}} = \left(\frac{d[\mathbf{X}]_{\mathcal{Y}}}{dt}\right)[\mathbf{X}]_{\mathcal{Y}}^{T}} \tag{3.5}$$

> We use a subscript $\mathcal{Y}$ because everything on the right hand side is expressed in $\mathcal{Y}$, so everything on the left hand side (i.e. $[\mathbf{W}]_{\mathcal{Y}}$) must be expressed in $\mathcal{Y}$ as well.

Solving for $d[\mathbf{X}]_{\mathcal{Y}}/dt$,

$$\frac{d[\mathbf{X}]_{\mathcal{Y}}}{dt} = [\mathbf{W}]_{\mathcal{Y}}\left([\mathbf{X}]_{\mathcal{Y}}^{T}\right)^{-1}$$

Since $[\mathbf{X}]_{\mathcal{Y}}$ is an orthogonal matrix, $[\mathbf{X}]_{\mathcal{Y}}^{-1} = [\mathbf{X}]_{\mathcal{Y}}^{T}$, so we have [110]

$$\boxed{\frac{d[\mathbf{X}]_{\mathcal{Y}}}{dt} = [\mathbf{W}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}} \tag{3.6}$$

> In general, we do not know $d[\mathbf{X}]_{\mathcal{Y}}/dt$, i.e. we do not know the time derivatives of the basis vectors of $\mathcal{X}$ expressed in $\mathcal{Y}$. However, for the case of *orthogonal matrices*, Eq (3.6) allows us to obtain this time derivative using two quantities measured at t instead: $[\mathbf{X}]_{\mathcal{Y}}(t)$ and $[\mathbf{W}]_{\mathcal{Y}}(t)$.

Substituting Eq. (3.6) into Eq. (3.4),

$$\begin{aligned}
\mathbf{0}_{n \times n} &= \left([\mathbf{W}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}\right)[\mathbf{X}]_{\mathcal{Y}}^{T} + [\mathbf{X}]_{\mathcal{Y}}\left([\mathbf{W}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}\right)^{T} \\
&= [\mathbf{W}]_{\mathcal{Y}}\left([\mathbf{X}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}^{T}\right) + [\mathbf{X}]_{\mathcal{Y}}\left([\mathbf{X}]_{\mathcal{Y}}^{T}[\mathbf{W}]_{\mathcal{Y}}^{T}\right) \\
&\quad [\mathbf{W}]_{\mathcal{Y}}\left([\mathbf{X}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}^{T}\right) + \left([\mathbf{X}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}^{T}\right)[\mathbf{W}]_{\mathcal{Y}}^{T} \\
&= [\mathbf{W}]_{\mathcal{Y}}\left(\mathbf{I}_{n \times n}\right) + \left(\mathbf{I}_{n \times n}\right)[\mathbf{W}]_{\mathcal{Y}}^{T}
\end{aligned}$$

$$\boxed{[\mathbf{W}]_{\mathcal{Y}} + [\mathbf{W}]_{\mathcal{Y}}^T = \mathbf{0}_{n \times n}}$$ (3.7)

Eq. (3.7) implies that $[\mathbf{W}]_{\mathcal{Y}}$ is a skew-symmetric matrix[1]; this is an essential property of $[\mathbf{W}]_{\mathcal{Y}}$ that is exploited in the three-dimensional case in Section 3.2.1. Now, substituting Eq. (3.6) into Eq. (3.3),

$$\frac{d[\mathbf{s}]_{\mathcal{Y}}}{dt} = \left([\mathbf{W}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}\right)[\mathbf{s}]_{\mathcal{X}} + [\mathbf{X}]_{\mathcal{Y}}\left(\frac{d[\mathbf{s}]_{\mathcal{X}}}{dt}\right)$$

$$\boxed{\frac{d[\mathbf{s}]_{\mathcal{Y}}}{dt} = [\mathbf{X}]_{\mathcal{Y}}\left(\frac{d[\mathbf{s}]_{\mathcal{X}}}{dt}\right) + [\mathbf{W}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}[\mathbf{s}]_{\mathcal{X}}}$$ (3.8)

3.3 Time Derivatives of Physical Vectors

Consider a physical vector, $\mathbf{s}$. Its coordinate vector expressed in some three-dimensional basis $B = (\hat{\mathbf{a}}, \hat{\mathbf{b}}, \hat{\mathbf{c}})$ is $[\mathbf{s}]_{\mathcal{B}} \in \mathbb{R}^3$. Our goal is to find its time derivative with respect to another three-dimensional basis $A = (\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$, where $\mathcal{B}$ is rotating with respect to $\mathcal{A}$, and where $\mathcal{A}$ and $\mathcal{B}$ same the origin.
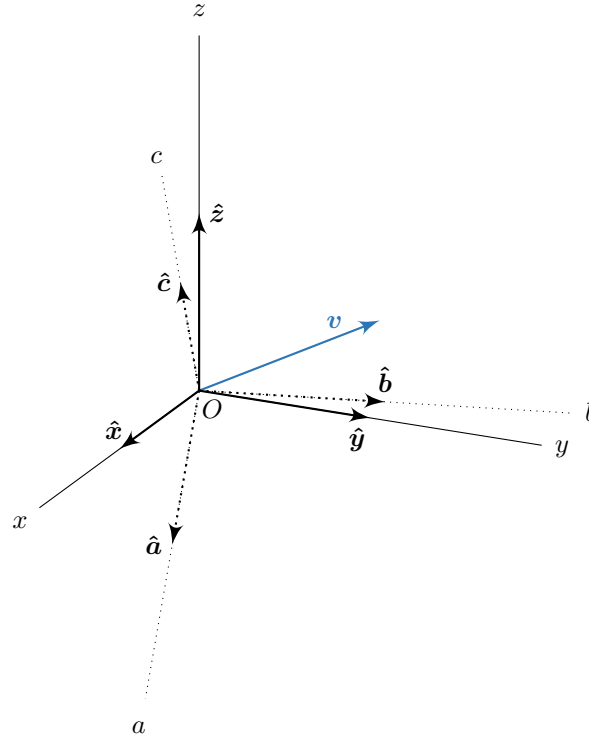
Figure 3.1: Physical vector with respect to two coordinate frames (with the same origin).

3.3.1 Time Derivative of the Coordinate Vector

Recall from Eq. (3.8) (repeated below for convenience) in Section 3.2.1 that the time derivative of an n-dimensional coordinate vector expressed in frame $\mathcal{Y}$ can be written in terms of the coordinate vector (and its time derivative)

---

[1] See Section **??**.

expressed in frame $\mathcal{X}$ as

$$\frac{d[\mathbf{s}]_{\mathcal{Y}}}{dt} = [\mathbf{X}]_{\mathcal{Y}} \left(\frac{d[\mathbf{s}]_{\mathcal{X}}}{dt} \right) + [\mathbf{W}]_{\mathcal{Y}}[\mathbf{X}]_{\mathcal{Y}}[\mathbf{s}]_{\mathcal{X}} \tag{3.8}$$

In this case, where $\mathcal{B}$ is rotating with respect to $\mathcal{A}$ and we want the time derivative with respect to $\mathcal{A}$, this equates to

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = [\boldsymbol{B}]_{\mathcal{A}} \left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt} \right) + [\boldsymbol{W}]_{\mathcal{A}}[\boldsymbol{B}]_{\mathcal{A}}[\boldsymbol{s}]_{\mathcal{B}}$$

where $[\boldsymbol{B}]_{\mathcal{A}}$ is the basis matrix of $\mathcal{B}$ expressed in $\mathcal{A}$ and $[\boldsymbol{W}]_{\mathcal{A}}$ is some matrix expressed in $\mathcal{A}$ that satisfies

$$[\boldsymbol{W}]_{\mathcal{A}} + [\boldsymbol{W}]_{\mathcal{A}}^{T} = \mathbf{0}_{n \times n}$$

Since we are dealing with the physical case, we restrict ourselves to $\mathbb{R}^3$. Let $[\boldsymbol{s}]_{\mathcal{A}} \in \mathbb{R}^3$ and $[\boldsymbol{s}]_{\mathcal{B}} \in \mathbb{R}^3$ be the coordinate vectors of the physical vector $\boldsymbol{s}$ expressed in coordinate frames $\mathcal{A}$ and $\mathcal{B}$, respectively. In this case, $[\boldsymbol{B}]_{\mathcal{A}}$ is just the rotation matrix $\mathbf{R}_{\mathcal{B} \to \mathcal{A}}$ (see Section **??**).

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = \mathbf{R}_{\mathcal{B} \to \mathcal{A}} \left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt} \right) + [\boldsymbol{W}]_{\mathcal{A}} \mathbf{R}_{\mathcal{B} \to \mathcal{A}}[\boldsymbol{s}]_{\mathcal{B}} \tag{3.9}$$

Additionally, in the the three-dimensional case, we have

$$[\boldsymbol{W}]_{\mathcal{A}} + [\boldsymbol{W}]_{\mathcal{A}}^{T} = \mathbf{0}_{3 \times 3}$$

Note that now, all the quantities inside the brackets are *physical* quantities and not mathematical quantities. However, we are still expressing these physical quantities in specific coordinate frames. The resulting "expressed" quantities are still mathematical vectors and matrices.

Eq. (3.9) is already extremely close to the end result in this section (you can skip forward and take a look at Eq. (3.13) if you would like). However, we can replace the matrix $[\boldsymbol{W}]_{\mathcal{A}}$ with a vector by recalling that it is a skew-symmetric matrix. Recall from Eq. (**??**) that for an arbitrary 3×3 skew-symmetric matrix $\mathbf{A}$,

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 0 & -a_{21} & a_{13} \\ a_{21} & 0 & -a_{32} \\ -a_{13} & a_{32} & 0 \end{bmatrix}$$

In the case of $[\boldsymbol{W}]_{\mathcal{A}}$, this implies that

$$[\boldsymbol{W}]_{\mathcal{A}} = \begin{bmatrix} w_{A,11} & w_{A,12} & w_{A,13} \\ w_{A,21} & w_{A,22} & w_{A,23} \\ w_{A,31} & w_{A,32} & w_{A,33} \end{bmatrix} = \begin{bmatrix} 0 & -w_{A,21} & w_{A,13} \\ w_{A,21} & 0 & -w_{A,32} \\ -w_{A,13} & w_{A,32} & 0 \end{bmatrix}$$

Let's define the physical vector $\boldsymbol{w}$. Expressing this vector in frame $\mathcal{A}$,

$$[\boldsymbol{w}]_{\mathcal{A}} = \begin{bmatrix} w_{A,32} \\ w_{A,13} \\ w_{A,21} \end{bmatrix}$$

implying that

$$\boxed{[\boldsymbol{w}]_{\mathcal{A}}^{\times} = [\boldsymbol{W}]_{\mathcal{A}}} \tag{3.10}$$

We can also write $[\boldsymbol{w}]_{\mathcal{A}}$ in terms of the rotation matrix $\mathbf{R}_{\mathcal{B} \to \mathcal{A}}$. We know from Eq. (3.5) that for this case, $[\boldsymbol{W}]_{\mathcal{A}}$ is defined as

$$[\boldsymbol{W}]_{\mathcal{A}} = \left(\frac{d\mathbf{R}_{\mathcal{B} \to \mathcal{A}}}{dt} \right) \mathbf{R}_{\mathcal{B} \to \mathcal{A}}^{T}$$

Substituting Eq. (3.10), it follows that

$$\boxed{[\boldsymbol{w}]_{\mathcal{A}}^{\times} = \left(\frac{d\mathbf{R}_{\mathcal{B} \to \mathcal{A}}}{dt} \right) \mathbf{R}_{\mathcal{B} \to \mathcal{A}}^{T}} \tag{3.11}$$

Now, let us return to the original problem at hand, which is finding the time derivative of a coordinate vector expressed in one frame in terms of a coordinate vector in another frame. Substituting Eq. (3.10) into Eq. (3.9),

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt}\right) + [\boldsymbol{w}]_{\mathcal{A}}^{\times}\mathbf{R}_{\mathcal{B}\to\mathcal{A}}[\boldsymbol{s}]_{\mathcal{B}} \tag{3.12}$$

Since $\mathbf{R}_{\mathcal{B}\to\mathcal{A}} \in \mathbb{R}^{3\times3}$ and $[\boldsymbol{s}]_{\mathcal{B}} \in \mathbb{R}^{3}$, we know that the product $\mathbf{R}_{\mathcal{B}\to\mathcal{A}}[\boldsymbol{s}]_{\mathcal{B}}$ is also a three-dimensional vector. Additionally, we know that for an arbitrary column vector $\mathbf{a} \in \mathbb{R}^{3}$,

$$[\boldsymbol{w}]_{\mathcal{A}}^{\times}\mathbf{a} = [\boldsymbol{w}]_{\mathcal{A}} \times \mathbf{a}$$

Substituting this into Eq. (3.12),

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt}\right) + [\boldsymbol{w}]_{\mathcal{A}} \times (\mathbf{R}_{\mathcal{B}\to\mathcal{A}}[\boldsymbol{s}]_{\mathcal{B}})$$

Finally, noting that $[\boldsymbol{w}]_{\mathcal{A}} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}[\boldsymbol{w}]_{\mathcal{A}}$,

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt}\right) + (\mathbf{R}_{\mathcal{B}\to\mathcal{A}}[\boldsymbol{w}]_{\mathcal{B}}) \times (\mathbf{R}_{\mathcal{B}\to\mathcal{A}}[\boldsymbol{s}]_{\mathcal{B}})$$

$$= \mathbf{R}_{\mathcal{B}\to\mathcal{A}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt}\right) + \mathbf{R}_{\mathcal{B}\to\mathcal{A}}\left([\boldsymbol{w}]_{\mathcal{B}} \times [\boldsymbol{s}]_{\mathcal{B}}\right)$$

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt} + [\boldsymbol{w}]_{\mathcal{B}} \times [\boldsymbol{s}]_{\mathcal{B}}\right) \tag{3.13}$$

> We do *not* box Eq. (3.13) because we still do not know exactly what $\boldsymbol{w}$ is. In Section 3.4, we will derive the angular velocity vector, and in Section 3.6, we will replace $\boldsymbol{w}$ with a particular angular velocity vector.

3.3.2   Time Derivative of the Physical Vector

Consider a physical vector $\boldsymbol{s}$ and a physical basis $A = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$. We can write $\boldsymbol{s}$ as

$$\boldsymbol{s} = A[\boldsymbol{s}]_{\mathcal{A}} = \begin{bmatrix} \hat{\boldsymbol{x}} & \hat{\boldsymbol{y}} & \hat{\boldsymbol{z}} \end{bmatrix} \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} = s_x\hat{\boldsymbol{x}} + s_y\hat{\boldsymbol{y}} + s_z\hat{\boldsymbol{z}}$$

Differentiating $\boldsymbol{s}$ with respect to time,

$$\frac{d\boldsymbol{s}}{dt} = \frac{d}{dt}\left(s_x\hat{\boldsymbol{x}} + s_y\hat{\boldsymbol{y}} + s_z\hat{\boldsymbol{z}}\right) = \left(\frac{ds_x}{dt}\hat{\boldsymbol{x}} + s_x\frac{d\hat{\boldsymbol{x}}}{dt}\right) + \left(\frac{ds_y}{dt}\hat{\boldsymbol{y}} + s_y\frac{d\hat{\boldsymbol{y}}}{dt}\right) + \left(\frac{ds_z}{dt}\hat{\boldsymbol{z}} + s_z\frac{d\hat{\boldsymbol{z}}}{dt}\right)$$

$$= \underbrace{\left(\frac{ds_x}{dt}\hat{\boldsymbol{x}} + \frac{ds_y}{dt}\hat{\boldsymbol{y}} + \frac{ds_z}{dt}\hat{\boldsymbol{z}}\right)}_{A\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt}} + \underbrace{\left(s_x\frac{d\hat{\boldsymbol{x}}}{dt} + s_y\frac{d\hat{\boldsymbol{y}}}{dt} + s_z\frac{d\hat{\boldsymbol{z}}}{dt}\right)}_{\substack{\text{ambiguous; what do we measure the rate} \\ \text{of change of the basis with respect to?}}}$$

We can immediately notice that this initial attempt to take the time derivative of a physical vector isn't quite complete. While we do obtain a portion that describes a vector's rate of change with respect to one basis, there is a second portion that is quite ambiguous; it involves rates of change of the *physical* basis vectors, which are abstract quantities. We have no way of quantifying these rates of change unless we define these rates to be **relative** to some other reference frame. Thus, we must clearly distinguish what frame we are taking a temporal derivative with respect to.

Let's consider taking the time derivative of $\boldsymbol{s}$ relative to frame $\mathcal{A}$. In terms of its coordinates expressed in frame $\mathcal{A}$, we can write $\boldsymbol{s}$ as

$$\boldsymbol{s} = s_x$$

We define the **time derivative of a physical vector** *relative to* a coordinate frame $\mathcal{B}$ as

$$\boxed{\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} = \boldsymbol{A}\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt}} \tag{3.14}$$

where $[\boldsymbol{s}]_{\mathcal{A}}$ is the coordinate vector of $\boldsymbol{s}$ expressed in frame $\mathcal{A}$. Equivalently, we can write

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{B}} = \boldsymbol{B}\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt}$$

At this point, we abandon the equation we started deriving at the beginning of this section, and instead consider Eq. (3.13) from Section 3.3.1, repeated below for convenience.

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt} + [\boldsymbol{w}]_{\mathcal{B}} \times [\boldsymbol{s}]_{\mathcal{B}}\right) \tag{3.13}$$

Left-multiplying both sides by the physical basis matrix, $\boldsymbol{A}$,

$$\underbrace{\boldsymbol{A}\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt}}_{\text{apply Eq. (3.14)}} = \underbrace{\boldsymbol{A}\mathbf{R}_{\mathcal{B}\to\mathcal{A}}}_{\substack{\boldsymbol{B} \\ \text{(by Eq. (2.23))}}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt} + [\boldsymbol{w}]_{\mathcal{B}} \times [\boldsymbol{s}]_{\mathcal{B}}\right)$$

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} = \boldsymbol{B}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt} + [\boldsymbol{w}]_{\mathcal{B}} \times [\boldsymbol{s}]_{\mathcal{B}}\right)$$

Distributing $\boldsymbol{B}$ on the right hand side,

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} = \underbrace{\boldsymbol{B}\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt}}_{\text{apply Eq. (3.14)}} + \boldsymbol{B}\left([\boldsymbol{w}]_{\mathcal{B}} \times [\boldsymbol{s}]_{\mathcal{B}}\right)$$

$$= \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{B}} + \boldsymbol{B}\left([\boldsymbol{w}]_{\mathcal{B}} \times [\boldsymbol{s}]_{\mathcal{B}}\right)$$

Finally, since $[\boldsymbol{w}]_{\mathcal{B}}$ and $[\boldsymbol{s}]_{\mathcal{B}}$ are both vectors expressed in frame $\mathcal{B}$, their cross product is also expressed in frame $\mathcal{B}$. Since this cross product will then be multiplied by the frame matrix $\boldsymbol{B}$ of frame $\mathcal{B}$, we can apply the concepts from Section 2.6.1 and say [83, p. 244]

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} = \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{B}} + \boldsymbol{w} \times \boldsymbol{s} \tag{3.15}$$

> Like Eq. (3.13), we do *not* box Eq. (3.15) because we still do not know exactly what $\boldsymbol{w}$ is. In Section 3.4, we will derive the angular velocity vector, and in Section 3.6, we will replace $\boldsymbol{w}$ with a particular angular velocity vector.

3.4   Angular Velocity

> While angular velocity may seem like something more applicable to attitude kinematics, it is actually essential both for translational *and* attitude kinematics. While we could introduce it in the next chapter on attitude kinematics, we choose to introduce it here for the following reasons:
>
>   1. The definition of the angular velocity is needed to define the *Golden Rule* (see

Section 3.6).

2. The ***Golden Rule*** is useful[a] for finding useful properties of the angular velocity and for deriving the angular acceleration.

3. Angular velocities/accelerations are *not* specific to attitude kinematics; they are needed in any case where there are rotating frames involved (for example, a point mass moving in a rotating frame). Instead, attitude kinematics is a subset of the use cases for angular velocities/accelerations.

---

[a] The ***Golden Rule*** is not strictly necessary to derive the properties of angular velocity or angular acceleration, but it does make it substantially easier.

Let us continue considering the coordinate frames $\mathcal{A} = (\hat{x}, \hat{y}, \hat{z})$ and $\mathcal{B} = (\hat{a}, \hat{b}, \hat{c})$, where $\mathcal{B}$ is rotating with respect to $\mathcal{A}$, and where the two frames share the same origin. We know we can express a vector r in frame $\mathcal{B}$ given its coordinate vector expressed in frame $\mathcal{A}$ as

$$[r]_B = \mathbf{R}_{\mathcal{A} \to \mathcal{B}} [r]_{\mathcal{A}}$$

where $\mathbf{R}_{\mathcal{A} \to \mathcal{B}}$ is the rotation matrix representing the passive rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$. In Section 4.4, we introduce the axis-angle representation where the rotation is parameterized as a single rotation about a single axis. The axis is defined using the **principal rotation vector**, $\mathbf{e}_{\mathcal{A} \to \mathcal{B}}$, and the angle of rotation is known as the **principal angle**, $\Phi_{\mathcal{A} \to \mathcal{B}}$. Note that in the axis-angle representation, $\mathbf{e}_{\mathcal{A} \to \mathcal{B}} = [\mathbf{e}_{\mathcal{A} \to \mathcal{B}}]_{\mathcal{A}} = [\mathbf{e}_{\mathcal{A} \to \mathcal{B}}]_B$ has the same coordinates whether it is expressed in frame $\mathcal{A}$ or in frame $\mathcal{B}$.

When considering angular velocity, it is convenient to think of the bases as functions of time, $\mathcal{A}(t)$ and $\mathcal{B}(t)$. Consider an infinitesimally small time interval, Δt. Since frame $\mathcal{A}$ is the stationary frame, we say

$$\mathcal{A}(t) = \mathcal{A}(t + \Delta t) = A$$

However, frame $\mathcal{B}$ is rotating. We say it has orientation B at time t, and orientation B' at time $t + \Delta t$.

$$\mathcal{B}(t) = B$$
$$\mathcal{B}(t + \Delta t) = B'$$

During this infinitesimally small time interval, frame $\mathcal{B}$ rotates from B to B'. We can use the axis-angle representation to express this rotation using an instantaneous axis of rotation, $\mathbf{e}$, and a small rotation about that axis, $\Delta \Phi$ (where Φ represents a general, i.e. not limited to small, rotation about $\mathbf{e}$).

$$\mathbf{e} = \text{principal rotation vector for rotation from } \mathcal{B} \text{ to } B'$$
$$\Delta \Phi = \text{principal angle about } \mathbf{e} \text{ for rotation from } \mathcal{B} \text{ to } B'$$

The representation of this rotation is visualized in Fig. 3.2. Note that

$$\mathbf{e} \neq \mathbf{e}_{\mathcal{A} \to \mathcal{B}}$$
$$\Delta \Phi \neq \Delta \Phi_{\mathcal{A} \to \mathcal{B}}$$

The approximate rate of change of the principal angle is

$$\frac{\Delta \Phi}{\Delta t}$$

and this rate of change occurs about the axis defined by $\mathbf{e}$. By the definition of the principal rotation vector, we know that

$$\mathbf{e} = [\mathbf{e}]_B = [\mathbf{e}]_{B'}$$

Let's consider the limit as $\Delta t \to 0$. In this limit,

$$\underbrace{\mathcal{B}(t)}_{B'} \approx \underbrace{\mathcal{B}(t + \Delta t)}_{B'}$$
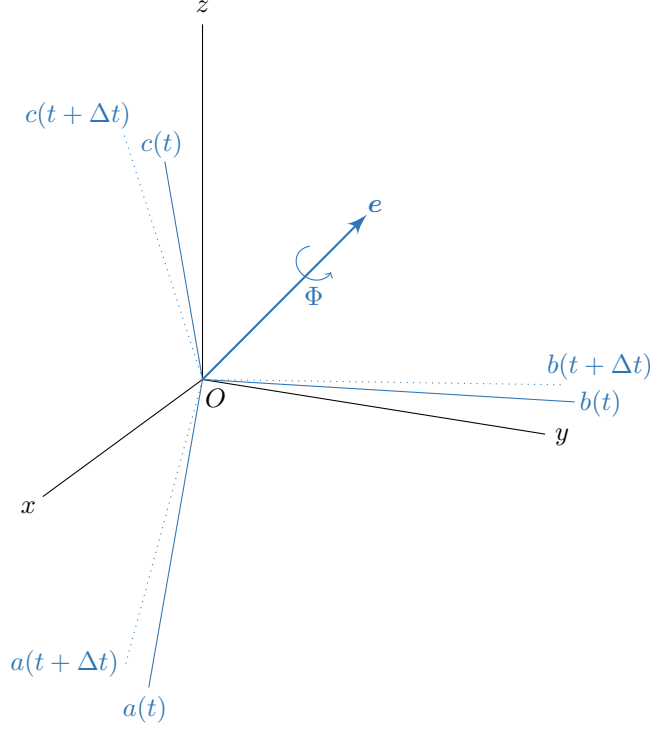
Figure 3.2: Instantaneous axis of rotation.

so we simply consider $\mathbf{e}$ expressed[2] in frame $\mathcal{B}$.

$$\mathbf{e} = [\mathbf{e}]_{\mathcal{B}}$$

We define the **angular velocity** of coordinate frame $\mathcal{B}$ with respect to (or relative to) coordinate frame $\mathcal{A}$, expressed in frame $\mathcal{A}$, as

$$\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} = \lim_{\Delta t \to 0} \left(\frac{\Delta \Phi}{\Delta t} [\mathbf{e}]_{\mathcal{B}} \right)$$

In the limit $\Delta t \to 0$, we also know that the instantaneous principal rotation vector, $[\mathbf{e}]_B$, is constant, so

$$\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} = \lim_{\Delta t \to 0} \left(\frac{\Delta \Phi}{\Delta t} \right) [\mathbf{e}]_{\mathcal{B}}$$

and we have that [95, pp. 30–31]

$$\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} = \frac{d\Phi}{dt} [\mathbf{e}]_{\mathcal{B}} \tag{3.16}$$

Thus, the angular velocity is a vector in the same direction as the instantaneous axis of rotation, as visualized in Fig. 3.3. 3.3.

---

[2]   Note that in Section 4.4, we do not explicitly denote the frame of the principal rotation vector; in that case, we simply had two frames $\mathcal{A}$ and $\mathcal{B}$ considered at the same instant in time, and the principal rotation vector had the same coordinates when expressed in either frame. However, in this case, we have three frames; $\mathcal{A}$ at time t, $\mathcal{B}$ at time t (B'), and $\mathcal{B}$ at time $t + \Delta t$. Since we are considering the rotation from B to B', $\mathbf{e}$ must be expressed in one of those two frames. Furthermore, since we end up considering the limit $\Delta t \to 0$, we just express $\mathbf{e}$ in frame $\mathcal{B}$.
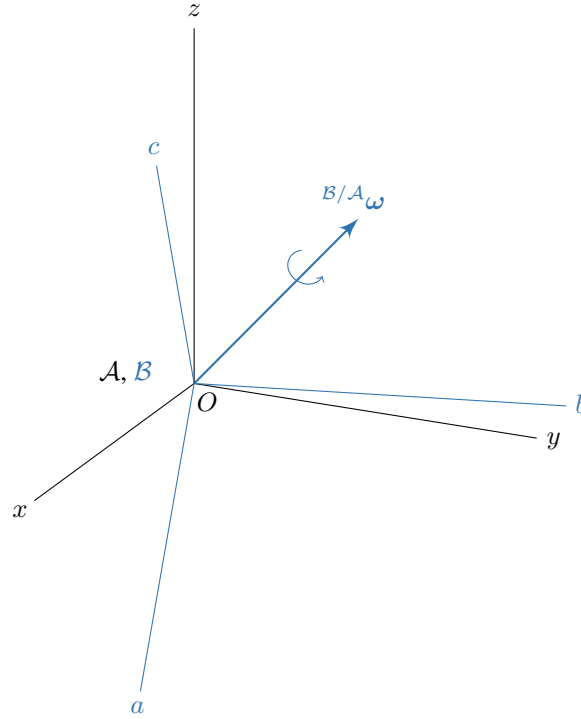
Figure 3.3: Angular velocity.

Convention 24: Angular velocity notation.

The angular velocity of coordinate frame $\mathcal{B}$ relative to coordinate frame $\mathcal{A}$ is denoted as

$$^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$$

Like other physical vectors, the angular velocity vector can be expressed in any coordinate frame. Expressing $^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$ in frame $\mathcal{B}$,

$$\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}}$$

While intuitively this definition of angular velocity makes sense (it represents a rate of rotation about some axis), it may not be obvious how this definition can be used in translational and/or attitude kinematics. However, this very same vector will arise both from taking the time derivative of a rotation matrix (see Section **??**) and from taking the time derivative of a quaternion (see Section 4.7).

3.4.1 Time Derivative of the Rotation Matrix

This section assumes that the reader is familiar with the axis-angle representation of a rotation, which will be discussed later in Section 4.4.

Let's continue our discussion regarding frames $\mathcal{A}$ and $\mathcal{B}$ (that share a common origin). Our goal in this section is to find the time derivative of $\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}$. Consider the following:

- frame $\mathcal{A}$ is the stationary frame and frame $\mathcal{B}$ is the rotating frame

- $\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t) = $ rotation matrix from $\mathcal{A}$ to $\mathcal{B}$ at time t
- $\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t+\Delta t) = $ rotation matrix from $\mathcal{A}$ to $\mathcal{B}$ at time $t+\Delta t$
- $\mathbf{R}_{\Delta t} = $ rotation matrix from attitude $\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)$ to $\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t+\Delta t)$

Since $\mathbf{R}_{\Delta t}$ is the rotation from attitude $\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)$ to $\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t+\Delta t)$, then we can recall from Section 2.5.1 that[3]

$$\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t+\Delta t) = \mathbf{R}_{\Delta t}\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t) \tag{3.17}$$

The time derivative of an arbitrary matrix $\mathbf{A}$ is defined as

$$\frac{d\mathbf{A}(t)}{dt} = \lim_{\Delta t\rightarrow 0} \frac{\mathbf{A}(t+\Delta t) - \mathbf{A}(t)}{\Delta t}$$

In this case,

$$\frac{d\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)}{dt} = \lim_{\Delta t\rightarrow 0} \frac{\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t+\Delta t) - \mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)}{\Delta t}$$

Substituting Eq. (3.17) into the equation above,

$$\frac{d\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)}{dt} = \lim_{\Delta t\rightarrow 0} \frac{\mathbf{R}_{\Delta t}\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t) - \mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)}{\Delta t} = \lim_{\Delta t\rightarrow 0} \frac{\left(\mathbf{R}_{\Delta t} - \mathbf{I}_{3\times 3}\right)\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)}{\Delta t}$$

Since $\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)$ is independent of Δt [95, pp. 30–31],

$$\frac{d\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)}{dt} = \lim_{\Delta t\rightarrow 0} \left(\frac{\mathbf{R}_{\Delta t} - \mathbf{I}_{3\times 3}}{\Delta t}\right)\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t) \tag{3.18}$$

To continue, we need to get a more useful form of $\mathbf{R}_{\Delta t}$. Recall that before, we parameterized the change of $\mathcal{B}$ from orientation B to B' using the axis-angle parameterization:

$$[\mathbf{e}]_{\mathcal{B}} = \text{principal rotation vector for rotation from } \mathcal{B} \text{ to } B'$$

$$\Delta\Phi = \text{principal angle about } [\mathbf{e}]_{\mathcal{B}} \text{ for rotation from } \mathcal{B} \text{ to } B'$$

Additionally, in Section 4.4.1, we introduce the following relationship between the axis-angle representation and the rotation matrix:

$$\mathbf{R} = \mathbf{I}_{3\times 3}(\cos\Phi) + (1 - \cos\Phi)\mathbf{e}\mathbf{e}^T - (\sin\Phi)\mathbf{e}^{\times}$$

In this case, we have

$$\mathbf{R}_{\Delta t} = \mathbf{I}_{3\times 3}(\cos\Delta\Phi) + (1 - \cos\Delta\Phi)[\mathbf{e}]_{\mathcal{B}}[\mathbf{e}]_{\mathcal{B}}^T - (\sin\Delta\Phi)[\mathbf{e}]_{\mathcal{B}}^{\times} \tag{3.19}$$

In the limiting case where $\Delta t \rightarrow 0$, we are considering a very small rotation (i.e. $\Delta\Phi \approx 0$). Thus, we make the small angle approximation [95, pp. 30, 43]

$$\sin\Delta\Phi \approx \Delta\Phi$$

$$\cos\Delta\Phi \approx 1$$

Applying this approximation to Eq. (3.19),

$$\mathbf{R}_{\Delta t} = \mathbf{I}_{3\times 3}\underbrace{(\cos\Delta\Phi)}_{\approx 1} + (1 - \underbrace{\cos\Delta\Phi}_{\approx 1})[\mathbf{e}]_{\mathcal{B}}[\mathbf{e}]_{\mathcal{B}}^T - \underbrace{(\sin\Delta\Phi)}_{\approx\Delta\Phi}[\mathbf{e}]_{\mathcal{B}}^{\times}$$

$$= \mathbf{I}_{3\times 3} - \Delta\Phi[\mathbf{e}]_{\mathcal{B}}^{\times}$$

Substituting this result back into Eq. (3.18),

$$\frac{d\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)}{dt} = \lim_{\Delta t\rightarrow 0} \left(\frac{\mathbf{I}_{3\times 3} - \Delta\Phi[\mathbf{e}]_{\mathcal{B}}^{\times} - \mathbf{I}_{3\times 3}}{\Delta t}\right)\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)$$

$$= -\left[\lim_{\Delta t\rightarrow 0}\left(\frac{\Delta\Phi[\mathbf{e}]_{\mathcal{B}}^{\times}}{\Delta t}\right)\right]\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}(t)$$

---

[3] Recall that to chain sequential rotations, we can simply use $\mathbf{R}_{A\rightarrow C} = \mathbf{R}_{B\rightarrow C}\mathbf{R}_{A\rightarrow B}$, where $\mathcal{A}$, $\mathcal{B}$, and C are arbitrary frames (see Section 2.5.1).

In the limiting case $\Delta t \to 0$, $[\mathbf{e}]_\mathcal{B}$ is considered stationary, and is not dependent on Δt. Thus, pulling it out from under the limit, we get

$$\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}(t)}{dt} = -\left[\lim_{\Delta t \to 0}\left(\frac{\Delta\Phi}{\Delta t}\right)[\mathbf{e}]_\mathcal{B}^\times\right]\mathbf{R}_{\mathcal{A}\to\mathcal{B}}(t)$$

$$= -\frac{d\Phi}{dt}[\mathbf{e}]_\mathcal{B}^\times\mathbf{R}_{\mathcal{A}\to\mathcal{B}}(t)$$

since $\Delta\Phi$ represents an infinitesimally small change in the rotation from frame $\mathcal{A}$ to frame $\mathcal{B}$. Substituting the limit with this derivative,

$$\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}(t)}{dt} = -\frac{d\Phi}{dt}[\mathbf{e}]_\mathcal{B}^\times\mathbf{R}_{\mathcal{A}\to\mathcal{B}}(t)$$

Since $d\Phi/dt$ is a scalar quantity, we can write[4]

$$\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}(t)}{dt} = -\left(\frac{d\Phi}{dt}[\mathbf{e}]_\mathcal{B}\right)^\times\mathbf{R}_{\mathcal{A}\to\mathcal{B}}(t)$$

By definition, the term in the parentheses is the angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ (see Eq. (3.16)) [95, pp. 30–31].

$$\boxed{\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}}{dt} = -\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_\mathcal{B}^\times\mathbf{R}_{\mathcal{A}\to\mathcal{B}}} \tag{3.20}$$

We go ahead and formalize this calculation as Algorithm 10 here.

Algorithm 10: `matderiv`
Time derivative of a passive rotation matrix.

Inputs:
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ — passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_\mathcal{B} \in \mathbb{R}^3$ — angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$

Procedure:
1. Obtain the skew-symmetric matrix form of $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_\mathcal{B}^\times$ (Algorithm 6).

$$\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_\mathcal{B}^\times = \texttt{vec2skew}\left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_\mathcal{B}\right)$$

2. Evaluate the time derivative of the passive rotation matrix from $\mathcal{A}$ to $\mathcal{B}$.

$$\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}}{dt} = -\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}^\times\right]_\mathcal{B}^\times\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$$

3. Return the result.

$$\textbf{return } \frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}}{dt}$$

Outputs:
- $\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}}{dt} \in \mathbb{R}^{3\times3}$ — time derivative of the passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$

Test Cases:
- See Appendix A.4.

[4] See Eq. (**??**) in Section **??**.

Alternate definitions of the angular velocity

Using Eq. (3.20), we can develop an alternate definition of the angular velocity. First, let's rearrange this equation slightly.

$$\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_B^{\times} \mathbf{R}_{\mathcal{A}\to\mathcal{B}} = -\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}}{dt}$$

Since rotation matrices are orthogonal matrices, $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T = \mathbf{I}_{3\times3}$. Right multiplying both sides by $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$,

$$\boxed{\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_B^{\times} = -\left(\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}}{dt}\right)\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T} \tag{3.21}$$

> Eq. (3.21) can be manipulated into many similar forms by noting the following:
>
> 1. $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}^T$ (Section **??**)
> 2. ${}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} = -{}^{\mathcal{A}/\mathcal{B}}\boldsymbol{\omega}$ (Section 3.4.2)
> 3. $\left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_B^{\times}\right)^T = -\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_B^{\times}$ (Section **??**)

Rotation matrix time derivative for the inverse rotation

Consider an arbitrary matrix $\mathbf{A}$. The derivative of the transpose is just the transpose of the derivative.

$$\frac{d(\mathbf{A}^T)}{dt} = \left(\frac{d\mathbf{A}}{dt}\right)^T$$

In this case,

$$\frac{d(\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T)}{dt} = \left(\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}}{dt}\right)^T$$

Since $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}$,

$$\frac{d\mathbf{R}_{\mathcal{B}\to\mathcal{A}}}{dt} = \left(\frac{d\mathbf{R}_{\mathcal{A}\to\mathcal{B}}}{dt}\right)^T \tag{3.22}$$

Notes on this derivation

The relationship between angular velocity and the rotation matrix time derivative is often given simply as

$$\frac{d\mathbf{R}}{dt} = \boldsymbol{\omega}^{\times}\mathbf{R}$$

Almost always, there is insufficient explanation regarding what rotation $\mathbf{R}$ represents, whether $\mathbf{R}$ represents an active or passive[5] rotation, and which angular velocity $\boldsymbol{\omega}$ describes (i.e. frame $\mathcal{B}$ with respect to $\mathcal{A}$, or the other way around) or which frame $\boldsymbol{\omega}$ is expressed in. Google searches will yield mixed results, and this is reflected in ChatGPT which changes its mind on if there's a negative sign or not every time you question it if its answer is correct[6].

---

[5]   Note that in this text, all rotations used in kinematics are passive.

[6]   This can be tested by asking it "What is the time derivative of the passive rotation matrix R_AB from frame A to frame B in terms of the angular velocity, w_AB, of frame B with respect to frame A, expressed in frame A?" Every time you ask it if the sign it uses is correct, it will change it's mind.

The most illuminating derivation I've found thus far for the time derivative of the rotation matrix is the one in [95, pp. 30–31, 43], which relies only on the most intuitive definition of the angular velocity (Eq. (3.16)) in terms of an axis-angle representation (which is also one of the more intuitive ways of visualizing a spatial rotation). Many derivations are derived from the perspective of physical vectors, which don't make it obvious which frame the angular velocity should be expressed in. Additionally, many consider the rotation of a physical vector relative to a stationary frame (i.e. an active rotation), instead of the rotation of a coordinate frame relative to a stationary vector (i.e. a passive rotation). Some additional resources for the time derivative of a rotation matrix are [105, pp. 511–512, pp. 514–515], [6], [83, pp. 283-284], [79], and [110].

[95, p. 31], [79], and [110] get the same result that we get in this section (although we should note that [79] just cites [110]). With the other references, the specifics of the rotations are less obvious (i.e. which frame vectors are expressed in, whether its a passive or active rotation, etc.). Due to the large amount of confusion and lack of clarity on this topic, it is useful to check our result by actually simulating some simple attitude kinematics. Let's consider a constant angular velocity

$$\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}^{\times} \right]_{\mathcal{B}} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \text{ rad/s}$$

Let the initial attitude of frame B with respect to frame A at time $t = 0$ be defined by the unit quaternion

$$\mathbf{q}_{\mathcal{A}\to\mathcal{B}}(0) = \frac{1}{\sqrt{1^2 + 2^2 + 3^2 + 4^2}} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

Verification via simulation

Due to the large amount of confusion on this topic, let's verify our mathematical results via a numerical solution. We can simulate the resulting attitude kinematics using the quaternion kinematic equation as computed by Algorithm 47 and a standard initial value problem (IVP) solver. The result will be a time series of quaternions, which we can convert to a time series of rotation matrices using Algorithm 39. We can find the time derivative of the rotation matrix at each instant using Algorithm 10. Additionally, we can approximate the derivative of each element of the rotation matrix by using a simple finite difference approximation. If Eq. (3.20) and Algorithm 10 are valid, then these derivatives should line up exactly. We plot the derivatives of R_{12} (an off-diagonal element was chosen to ensure we didn't mess up a transpose) found using Algorithm 10 and using numerical differentiation in Fig. 3.4 below.

> The full source code for this example is included in Appendix **??**.

3.4.2 Properties

> This section skips ahead slightly and makes use of the ***Golden Rule*** discussed in Section 3.6.

Additive property

Consider an arbitrary physical vector $\boldsymbol{s}$, and three coordinate frames A, B, and C. Using Eq. (**??**), we can write the following three equations:

$$\left. \frac{d\boldsymbol{s}}{dt} \right|_{B} = \left. \frac{d\boldsymbol{s}}{dt} \right|_{\mathcal{A}} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{s} \tag{3.23}$$
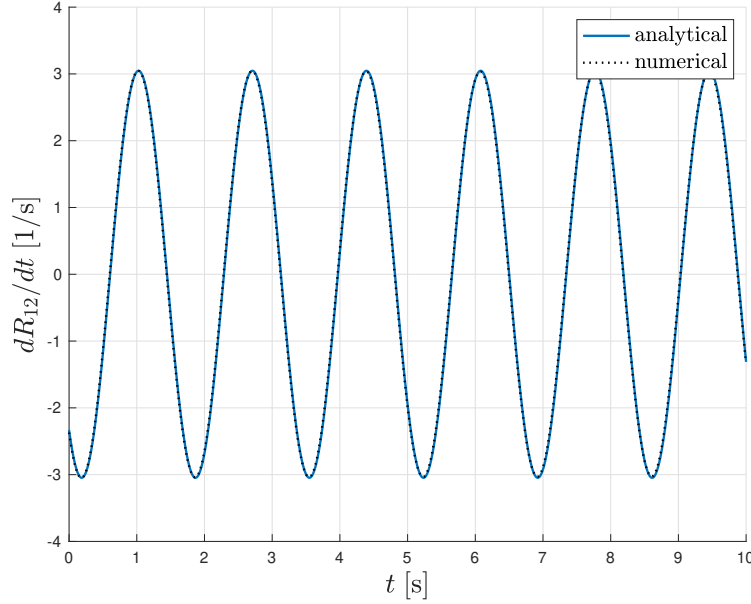
Figure 3.4: Time derivative of $(1, 2)$ element of $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$ using both Algorithm 10 and a numerical approximation.

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_C = \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} + {}^{C/A}\boldsymbol{\omega} \times \boldsymbol{s} \tag{3.24}$$

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_C = \left.\frac{d\boldsymbol{s}}{dt}\right|_{B} + {}^{C/B}\boldsymbol{\omega} \times \boldsymbol{s} \tag{3.25}$$

Substituting Eq. (3.23) into Eq. (3.25),

$$\begin{aligned}
\left.\frac{d\boldsymbol{s}}{dt}\right|_C &= \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} + \left({}^{B/A}\boldsymbol{\omega} \times \boldsymbol{s}\right) + \left({}^{C/B}\boldsymbol{\omega} \times \boldsymbol{s}\right) \\
&= \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} + \left({}^{B/A}\boldsymbol{\omega} + {}^{C/B}\boldsymbol{\omega}\right) \times \boldsymbol{s}
\end{aligned} \tag{3.26}$$

Substituting the right hand side of Eq. (3.24) into the left hand side of (3.26),

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} + {}^{C/A}\boldsymbol{\omega} \times \boldsymbol{s} = \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} + \left({}^{B/A}\boldsymbol{\omega} + {}^{C/B}\boldsymbol{\omega}\right) \times \boldsymbol{s}$$

$$\left({}^{C/A}\boldsymbol{\omega}\right) \times \boldsymbol{s} = \left({}^{B/A}\boldsymbol{\omega} + {}^{C/B}\boldsymbol{\omega}\right) \times \boldsymbol{s}$$

It follows that [46, p. 10]

$$\boxed{{}^{C/A}\boldsymbol{\omega} = {}^{B/A}\boldsymbol{\omega} + {}^{C/B}\boldsymbol{\omega}} \tag{3.27}$$

We refer to this property as the **additive property** since it implies that the angular velocity is *additive* over multiple frames [91, p. 18].

In the practical implementation of Eq. (3.27), we must consider which frames the vectors are expressed in. Typically, we will have ${}^{B/A}\boldsymbol{\omega}$ expressed in frame $\mathcal{B}$, and ${}^{C/B}\boldsymbol{\omega}$ expressed in frame $\mathcal{C}$. Additionally, we will want ${}^{C/A}\boldsymbol{\omega}$ expressed in frame $\mathcal{C}$. Expressing both sides of Eq. (3.27) in frame $\mathcal{C}$,

$$\boxed{\left[{}^{C/A}\boldsymbol{\omega}\right]_{\mathcal{C}} = \mathbf{R}_{\mathcal{B}\to\mathcal{C}}\left[{}^{B/A}\boldsymbol{\omega}\right]_{\mathcal{B}} + \left[{}^{C/B}\boldsymbol{\omega}\right]_{\mathcal{C}}} \tag{3.28}$$

Algorithm 11: `add_ang_vel`

Addition of angular velocities.

Inputs:

- $\left[{}^{B/A}\boldsymbol{\omega}\right]_B \in \mathbb{R}^3$ - angular velocity of frame B with respect to frame A, expressed in frame B [rad/s]
- $\left[{}^{C/B}\boldsymbol{\omega}\right]_C \in \mathbb{R}^3$ - angular velocity of frame C with respect to frame B, expressed in frame C [rad/s]
- $\mathbf{R}_{B \to C} \in \mathbb{R}^{3 \times 3}$ - passive rotation matrix from frame B to frame C

Procedure:

$$\left[{}^{C/A}\boldsymbol{\omega}\right]_C = \mathbf{R}_{B \to C}\left[{}^{B/A}\boldsymbol{\omega}\right]_B + \left[{}^{C/B}\boldsymbol{\omega}\right]_C$$

Outputs:

- $\left[{}^{C/A}\boldsymbol{\omega}\right]_C \in \mathbb{R}^3$ - angular velocity of frame C with respect to frame A, expressed in frame C [rad/s]

Test Cases:

- See Appendix A.4.

Relative property

Using Eq. (3.27), we can write

$${}^{B/A}\boldsymbol{\omega} + {}^{A/B}\boldsymbol{\omega} = {}^{A/A}\boldsymbol{\omega}$$

by replacing C with A. By definition, the angular velocity of a frame with respect to itself is 0. Therefore, we know that ${}^{A/A}\boldsymbol{\omega} = \mathbf{0}$, so

$${}^{B/A}\boldsymbol{\omega} + {}^{A/B}\boldsymbol{\omega} = \mathbf{0}$$

It follows that

$$\boxed{{}^{B/A}\boldsymbol{\omega} = -\left({}^{A/B}\boldsymbol{\omega}\right)} \tag{3.29}$$

We refer to this property as the **relative property** since it implies that the angular velocity of frame B relative to frame A is simply the opposite of the angular velocity of frame A relative to frame B [91, p. 18].

Derivative property

Let's take the time derivative of ${}^{B/A}\boldsymbol{\omega}$ relative to frame B. From Eq. (**??**),

$$\left.\frac{d\left({}^{B/A}\boldsymbol{\omega}\right)}{dt}\right|_B = \left.\frac{d\left({}^{B/A}\boldsymbol{\omega}\right)}{dt}\right|_A + \underbrace{{}^{B/A}\boldsymbol{\omega} \times {}^{B/A}\boldsymbol{\omega}}_{=\mathbf{0}}$$

$$\boxed{\left.\frac{d\left({}^{B/A}\boldsymbol{\omega}\right)}{dt}\right|_B = \left.\frac{d\left({}^{B/A}\boldsymbol{\omega}\right)}{dt}\right|_A} \tag{3.30}$$

We refer to this property as the **derivative property** since it implies that the time derivative of the angular velocity of frame B relative to frame A is identical whether you take the derivative in frame A or frame B [46, p. 29], [91, p. 18].

3.4.3 Spin vs. Orbital Angular Velocity

In the study of mechanics, the **spin angular velocity** describes how fast a rigid body rotates with respect to its center of origin, and is independent of the choice of origin. More generally, we can consider the spin angular velocity as the angular velocity of one coordinate frame with respect to another. In the context of mechanics, there is a coordinate frame, called the body frame, fixed to the rigid body, and we examine the angular velocity of the body frame with respect to a world frame. Note that the body frame and world frame do *not* have the same origin; the spin angular velocity just compares how the directions of the coordinate axes rotate with respect to one another. For more information on the world and body frames, see Section 4.1 in the next chapter.

So far in this chapter, the angular velocities we have considered are spin angular velocities. However, there is another type of angular velocity, called the **orbital angular velocity**, which refers to how fast a particle orbits about a fixed origin. While the spin and orbital angular velocities are often thought of as completely different concepts, the orbital angular velocity is actually a special case of the spin angular velocity; we cover this sub-case in depth in Section **??** later in this chapter. As a rule of thumb, we intuit an angular velocity as:

- a *spin* angular velocity for rigid bodies undergoing rotation motion, and
- as an *orbital* angular velocity for a particle orbiting a fixed point

We will see in Section **??** that these two types of angular velocity are fundamentally the same thing.

3.5 Angular Acceleration

> This section skips ahead slightly and makes use of the *Golden Rule* and the frame derivative, discussed in Sections 3.6 and 3.7, respectively.

The **angular acceleration** of coordinate frame $\mathcal{B}$ with respect to coordinate frame $\mathcal{A}$ is simply the time derivative of the angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, taken in either frame.

$$\boxed{{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} = \frac{d\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right)}{dt}\bigg|_{\mathcal{B}} = \frac{d\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right)}{dt}\bigg|_{\mathcal{A}}} \tag{3.31}$$

The fact that we can take the time derivative relative to either frame comes directly from the derivative property of the angular velocity (Eq. (3.30) in Section 3.4.2) [46, pp. 28–29].

Additive identity

Unlike the angular velocity, the angular acceleration is ***not*** additive over frames [91, p. 18].

$$ {}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\alpha} \neq {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} + {}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha} $$

Therefore, while the angular velocity had an additive *property*, we refer to the analogous equation of the angular acceleration as the **additive *identity***.

To derive how angular accelerations between different frames are related, recall the additive property for angular velocities given by Eq. (3.27).

$$ {}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\omega} = {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} + {}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega} \tag{3.27}$$

Taking the frame derivative of Eq. (3.27) relative to frame $\mathcal{A}$,

$$ \frac{d\left({}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\omega}\right)}{dt}\bigg|_{\mathcal{A}} = \frac{d\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right)}{dt}\bigg|_{\mathcal{A}} + \frac{d\left({}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right)}{dt}\bigg|_{\mathcal{A}} $$

By applying the definition of the angular acceleration from Eq. (3.31), we can immediately replace the first two derivatives with angular accelerations.

$$
{}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\alpha} = {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} + \left.\frac{d\left({}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right)}{dt}\right|_{\mathcal{A}} \tag{3.32}
$$

Using either the definition of the frame derivative (Eq. (3.42) from Section 3.7) or the ***Golden Rule*** (Eq. (3.40) from Section 3.6), we can rewrite the remaining derivative term as

$$
\left.\frac{d\left({}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right)}{dt}\right|_{\mathcal{A}} = \underbrace{\left.\frac{d\left({}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right)}{dt}\right|_{\mathcal{B}}}_{{}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha}} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}
$$

$$
\left.\frac{d\left({}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right)}{dt}\right|_{\mathcal{A}} = {}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega} \tag{3.33}
$$

Substituting Eq. (3.33) into Eq. (3.32) [83, p. 246],

$$
\boxed{{}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\alpha} = {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} + {}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}} \tag{3.34}
$$

In the practical implementation of Eq. (**??**), we must consider which frames the vectors are expressed in. Typically, we will have ${}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$ and ${}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}$ expressed in frame $\mathcal{B}$, and ${}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}$ and ${}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha}$ expressed in frame $\mathcal{C}$. Additionally, we will want ${}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\alpha}$ expressed in frame $\mathcal{C}$. Expressing both sides of Eq. (3.34) in frame $\mathcal{C}$,

$$
\boxed{\left[{}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{C}} = \mathbf{R}_{\mathcal{B}\to\mathcal{C}}\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} + \left[{}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha}\right]_{\mathcal{C}} + \mathbf{R}_{\mathcal{B}\to\mathcal{C}}\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[{}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right]_{\mathcal{C}}} \tag{3.35}
$$

Algorithm 12: `add_ang_acc`
Addition of angular accelerations.

Inputs:
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — angular acceleration of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ $[\mathrm{rad/s^2}]$
- $\left[{}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right]_{\mathcal{C}} \in \mathbb{R}^3$ — angular velocity of frame $\mathcal{C}$ with respect to frame $\mathcal{B}$, expressed in frame $\mathcal{C}$ $[\mathrm{rad/s}]$
- $\left[{}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha}\right]_{\mathcal{C}} \in \mathbb{R}^3$ — angular acceleration of frame $\mathcal{C}$ with respect to frame $\mathcal{B}$, expressed in frame $\mathcal{C}$ $[\mathrm{rad/s^2}]$
- $\mathbf{R}_{\mathcal{B}\to\mathcal{C}} \in \mathbb{R}^{3\times3}$ — passive rotation matrix from frame $\mathcal{B}$ to frame $\mathcal{C}$

Procedure:

$$
\left[{}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{C}} = \mathbf{R}_{\mathcal{B}\to\mathcal{C}}\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} + \left[{}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha}\right]_{\mathcal{C}} + \mathbf{R}_{\mathcal{B}\to\mathcal{C}}\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[{}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right]_{\mathcal{C}}
$$

Outputs:
- $\left[{}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{C}} \in \mathbb{R}^3$ — angular acceleration of frame $\mathcal{C}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{C}$ $[\mathrm{rad/s^2}]$

Test Cases:
- See Appendix A.4.

Relative property

Using Eq. (3.34), we can write

$$^{A/A}\boldsymbol{\alpha} = {}^{B/A}\boldsymbol{\alpha} + {}^{A/B}\boldsymbol{\alpha} + {}^{B/A}\boldsymbol{\omega} \times {}^{A/B}\boldsymbol{\omega}$$

by replacing $\mathcal{C}$ with $\mathcal{A}$. By definition, the angular acceleration of a frame with respect to itself is 0. Therefore, we know that $^{A/A}\boldsymbol{\alpha} = \mathbf{0}$, so

$$\mathbf{0} = {}^{B/A}\boldsymbol{\alpha} + {}^{A/B}\boldsymbol{\alpha} + {}^{B/A}\boldsymbol{\omega} \times {}^{A/B}\boldsymbol{\omega}$$

Additionally, from the relative property for angular velocities, we know that $^{B/A}\boldsymbol{\omega} = -\,{}^{A/B}\boldsymbol{\omega}$. Substituting this into the equation above,

$$\mathbf{0} = {}^{B/A}\boldsymbol{\alpha} + {}^{A/B}\boldsymbol{\alpha} + {}^{B/A}\boldsymbol{\omega} \times \left(-\,{}^{B/A}\boldsymbol{\omega}\right)$$

Since $^{B/A}\boldsymbol{\omega}$ and $-\,{}^{B/A}\boldsymbol{\omega}$ are antiparallel, their cross product is 0, and we have

$$\mathbf{0} = {}^{B/A}\boldsymbol{\alpha} + {}^{A/B}\boldsymbol{\alpha}$$

It follows that

$$\boxed{{}^{B/A}\boldsymbol{\alpha} = -\left({}^{A/B}\boldsymbol{\alpha}\right)} \tag{3.36}$$

We refer to this property as the **relative property** since it implies that the angular acceleration of frame B relative to frame A is simply the opposite of the angular acceleration of frame A relative to frame B. Note that this relative property for angular accelerations is analogous to the relative property for angular velocities.

3.6 The *Golden Rule* of Kinematics

Recall Eqs. (3.13) (derived in Section **??**) and (3.15) (derived in Section **??**), repeated below for convenience.

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = \mathbf{R}_{\mathcal{B} \to \mathcal{A}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt}\right) + [\boldsymbol{w}]_{\mathcal{A}} \times \left(\mathbf{R}_{\mathcal{B} \to \mathcal{A}}[\boldsymbol{s}]_{\mathcal{B}}\right) \tag{3.13}$$

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} = \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{B}} + \boldsymbol{w} \times \boldsymbol{s} \tag{3.15}$$

At the time these equations were derived, we did not know exactly what this vector $\boldsymbol{w}$ represented, albeit we *did* know it was some quantity associated with rotational kinematics. The aim of this section is to replace the unknown $\boldsymbol{w}$ with a particular angular velocity. When we do so, these equations become definitions that are fundamental to both translational and attitude (rotational) kinematics. There is no "standard" name for these equations; some names include the *transport theorem*, *transport equation*, *rate of change transport theorem*, *basic kinematic equation*, and *Euler derivative transformation formula* [98]. However, doing a Google search of these names will often instead yield the transport theorem for control volumes, or Euler's equation of motion for attitude dynamics. In this text, we refer to these equations as the *Golden Rule* to signify their importance to the study of kinematics[7].

To begin, recall Eq. (3.11) (from Section **??**), where we wrote $[\boldsymbol{w}]_{\mathcal{A}}$ in terms of the rotation matrix $\mathbf{R}_{\mathcal{B} \to \mathcal{A}}$ (and its time derivative) as

$$[\boldsymbol{w}]_{\mathcal{A}}^{\times} = \left(\frac{d\mathbf{R}_{\mathcal{B} \to \mathcal{A}}}{dt}\right)\mathbf{R}_{\mathcal{B} \to \mathcal{A}}^{T} \tag{3.11}$$

Recall from Eq. (3.21) (in Section **??**) that

$$\left[{}^{B/A}\boldsymbol{\omega}\right]_{B}^{\times} = -\left(\frac{d\mathbf{R}_{\mathcal{A} \to \mathcal{B}}}{dt}\right)\mathbf{R}_{\mathcal{A} \to \mathcal{B}}^{T}$$

---

[7] This name is used in Stanford's classical dynamics course (AA 242A).

Switching frame $\mathcal{A}$ with $\mathcal{B}$,

$$\left[^{\mathcal{A}/\mathcal{B}}\boldsymbol{\omega}\right]_{\mathcal{A}}^{\times} = -\left(\frac{d\mathbf{R}_{\mathcal{B}\to\mathcal{A}}}{dt}\right)\mathbf{R}_{\mathcal{B}\to\mathcal{A}}^{T}$$

From Section 3.4.2, we know that

$$^{\mathcal{A}/\mathcal{B}}\boldsymbol{\omega} = -\,^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$$

Thus,

$$\left[-\,^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{A}}^{\times} = -\left(\frac{d\mathbf{R}_{\mathcal{B}\to\mathcal{A}}}{dt}\right)\mathbf{R}_{\mathcal{B}\to\mathcal{A}}^{T} \quad \to \quad -\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{A}}^{\times} = -\left(\frac{d\mathbf{R}_{\mathcal{B}\to\mathcal{A}}}{dt}\right)\mathbf{R}_{\mathcal{B}\to\mathcal{A}}^{T}$$

$$\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{A}}^{\times} = \left(\frac{d\mathbf{R}_{\mathcal{B}\to\mathcal{A}}}{dt}\right)\mathbf{R}_{\mathcal{B}\to\mathcal{A}}^{T}$$

Comparing this to Eq. (3.11),

$$[\boldsymbol{w}]_{\mathcal{A}}^{\times} = \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{A}}^{\times}$$

and it follows that

$$\boxed{\boldsymbol{w} = \,^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}} \tag{3.37}$$

Immediately, we can replace the $\boldsymbol{w}$'s in Eqs. (3.13) and (3.15) with $^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$.

$$\frac{d[\boldsymbol{s}]_{\mathcal{A}}}{dt} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}\left(\frac{d[\boldsymbol{s}]_{\mathcal{B}}}{dt}\right) + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{A}} \times (\mathbf{R}_{\mathcal{B}\to\mathcal{A}}[\boldsymbol{s}]_{\mathcal{B}}) \tag{3.38}$$

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} = \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{B}} + \,^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{s} \tag{3.39}$$

However, we do not yet box Eqs. (3.38) or (3.39). To generalize these equations further, we must introduce the concept of rate vectors, which we cover shortly in Section 3.6.1. Section 3.6.2 will then be focused on the physical form of the *Golden Rule* while Section 3.6.3 will be focused on its coordinate form.

3.6.1 Rate Vectors

In Eq. (**??**), we just defined the time derivative of a physical vector relative to coordinate frame $\mathcal{B}$ can be written in terms of its time derivative relative to coordinate frame $\mathcal{A}$ as

$$\left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{A}} = \left.\frac{d\boldsymbol{s}}{dt}\right|_{\mathcal{B}} + \,^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{s}$$

However, this relationship is still missing a critical component; it completely neglects the fact that $\boldsymbol{s}$ may *itself* be a vector representing a time derivative of another physical vector relative to some coordinate frame.

Let $\boldsymbol{s}$ be a physical vector that is defined as the time derivative of some other physical vector $\boldsymbol{x}$. Then $\boldsymbol{s}$ is a **rate vector**, and we should denote which frame the rate of change of $\boldsymbol{x}$ is measured relative to.

> **Convention 25:** Notation for rate vectors.
>
> If $\boldsymbol{s}$ is a rate vector that defines the time rate of change of another vector relative to coordinate frame $\mathcal{A}$, then we denote it as
>
> $$^{\mathcal{A}}\boldsymbol{s}$$

For example, if we had $^{\mathcal{B}}\boldsymbol{s}$, then using Eq. (3.15) we could write its time derivative relative to frame $\mathcal{A}$ as

$$\left.\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\right|_{\mathcal{A}} = \left.\frac{d(^{\mathcal{B}}\boldsymbol{s})}{dt}\right|_{\mathcal{B}} + \,^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \,^{\mathcal{B}}\boldsymbol{s}$$

Note that rate vectors are still just physical vectors and have all the same properties. For example, we can express $^{\mathcal{B}}\boldsymbol{s}$ in frame $\mathcal{B}$ as $[^{\mathcal{B}}\boldsymbol{s}]_{\mathcal{B}}$. Additionally, from Eq. (3.14), we also have

$$\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{B}} = \boldsymbol{B}\frac{d\left[^{\mathcal{B}}\boldsymbol{s}\right]_{\mathcal{B}}}{dt}$$

We can denote the magnitude of a rate vector in the same we would with a standard physical vector, but including the frame label.

$$\left|^{A}\boldsymbol{s}\right| = {}^{A}s$$

> **Convention 26:** Notation for the magnitude of a rate vector.
>
> Since the magnitude of a physical vector is a physical scalar, it is written using the italic, non-boldface version of the same symbol. For example, the magnitude of $^{A}\boldsymbol{s}$ is written as
>
> $$^{A}s$$

3.6.2   Physical Form of the *Golden Rule*

Consider two coordinate frames, $\mathcal{A} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$ and $\mathcal{B} = (\hat{\boldsymbol{a}}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{c}})$, with the same origin O, where $\mathcal{B}$ is rotating with respect to $\mathcal{B}$ with angular velocity $^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$. Consider an arbitrary physical vector, $^{\mathcal{B}}\boldsymbol{s}$, which is a rate vector relative to frame $\mathcal{B}$. If we know $^{\mathcal{B}}\boldsymbol{s}$ and its time derivative relative to frame $\mathcal{B}$, then we can find its time derivative relative to frame $\mathcal{A}$ as [83, pp. 245–247]

$$\boxed{\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{A}} = \frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{B}} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{s}} \tag{3.40}$$

Note that the first term represents the contribution from the rate of change of $^{\mathcal{B}}\boldsymbol{s}$ with respect to frame $\mathcal{B}$, while the second term represents the contribution from the rotation of frame $\mathcal{B}$ relative to frame $\mathcal{A}$.

$$\frac{d\left(^{\mathcal{A}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{A}} = \underbrace{\frac{d\left(^{\mathcal{A}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{B}}}_{\substack{\text{contribution from} \\ \text{rate of change} \\ \text{w.r.t. frame } \mathcal{B}}} + \underbrace{{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{A}}\boldsymbol{s}}_{\substack{\text{contribution from} \\ \text{rotation of frame } \mathcal{B} \\ \text{relative to frame } \mathcal{A}}}$$

3.6.3   Coordinate Form of the *Golden Rule*

We can use Eq. (**??**) to develop equations that describe the motion and vector properties of physical bodies. However, to simulate kinematics, we need the same equation but in a matrix algebra form in terms of coordinate vectors and matrices. Typically, we will have everything on the right hand side of Eq. (**??**) expressed in frame $\mathcal{B}$. Therefore, expressing both sides in frame $\mathcal{B}$,

$$\left[\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{A}}\right]_{\mathcal{B}} = \left[\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{B}}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}\boldsymbol{s}\right]_{\mathcal{B}}$$

However, we will also typically want the derivative term on the LHS expressed in frame $\mathcal{A}$. We can write

$$\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}\left[\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{A}}\right]_{\mathcal{A}} = \left[\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{B}}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}\boldsymbol{s}\right]_{\mathcal{B}}$$

Left-multiplying both sides by $\mathbf{R}^{T}_{\mathcal{A}\rightarrow\mathcal{B}}$, and noting that $\mathbf{R}^{T}_{\mathcal{A}\rightarrow\mathcal{B}}\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}} = \mathbf{I}_{3\times3}$,

$$\boxed{\left[\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{A}}\right]_{\mathcal{A}} = \mathbf{R}^{T}_{\mathcal{A}\rightarrow\mathcal{B}}\left\{\left[\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{B}}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}\boldsymbol{s}\right]_{\mathcal{B}}\right\}} \tag{3.41}$$

Eq. $(3.41)^8$ represents the **coordinate form of the** *Golden Rule*. We *could* simplify this equation slightly further by noting that $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^{T} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}$, but in most practical situations, we will actually have knowledge of $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$.

3.7 The Frame Derivative

The *Golden Rule* from the previous section is one of the fundamental relationships in kinematics. However, it is also quite rigid in its application. When applying the *Golden Rule* it is much easier to think of it as an operator.

Consider the standard time derivative operator, d/dt, operating on some mathematical quantity (i.e. scalar, column vector, matrix, etc.).

$$\frac{d}{dt}(\cdot) = \frac{d(\cdot)}{dt}$$

As an example, consider an arbitrary mathematical vector, $\boldsymbol{s} \in \mathbb{R}^n$. If we apply the time derivative operator $\boldsymbol{s}$, we obtain the time derivative of $\boldsymbol{s}$.

$$\frac{d}{dt}(\boldsymbol{s}) = \frac{d\boldsymbol{s}}{dt}$$

For physical vectors, we cannot simply take the time derivative; the rate of change of a physical vector is always relative to some coordinate frame (i.e. some frame of reference). We define the **frame derivative operator** as [86, p. XI], [77]

$$\left.\frac{d}{dt}\right|_{\mathcal{A}}(\cdot) = \left.\frac{d}{dt}\right|_{\mathcal{B}}(\cdot) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times (\cdot) \tag{3.42}$$

As an example, consider an arbitrary physical vector, ${}^{\mathcal{B}}\boldsymbol{s}$, which itself is a rate vector relative to frame $\mathcal{B}$. The **frame derivative** of ${}^{\mathcal{B}}\boldsymbol{s}$ relative to frame $\mathcal{A}$ is simply the time derivative of ${}^{\mathcal{B}}\boldsymbol{s}$ relative to (i.e. as viewed from) frame $\mathcal{A}$; we can find this frame derivative by applying the frame derivative operator.

$$\left.\frac{d}{dt}\right|_{\mathcal{A}}\left({}^{\mathcal{B}}\boldsymbol{s}\right) = \left.\frac{d}{dt}\right|_{\mathcal{B}}\left({}^{\mathcal{B}}\boldsymbol{s}\right) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left({}^{\mathcal{B}}\boldsymbol{s}\right)$$

$$\left.\frac{d\left({}^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\right|_{\mathcal{A}} = \left.\frac{d\left({}^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\right|_{\mathcal{B}} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{s} \tag{3.43}$$

Note that this equation is identical to the *Golden Rule* given by Eq. (**??**).

Like the standard derivative, the frame derivative is a linear operator. To show this, consider the vector $c_1\boldsymbol{s}_1 + c_2\boldsymbol{s}_2$, where $c_1, c_2 \in \mathbb{R}$ are scalars and $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ are physical vectors. Applying the frame derivative operator,

$$\left.\frac{d}{dt}\right|_{\mathcal{A}}(c_1\boldsymbol{s}_1 + c_2\boldsymbol{s}_2) = \left.\frac{d}{dt}\right|_{\mathcal{B}}(c_1\boldsymbol{s}_1 + c_2\boldsymbol{s}_2) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times (c_1\boldsymbol{s}_1 + c_2\boldsymbol{s}_2)$$

$$= \left.\frac{d}{dt}\right|_{\mathcal{B}}(c_1\boldsymbol{s}_1) + \left.\frac{d}{dt}\right|_{\mathcal{B}}(c_2\boldsymbol{s}_1) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times (c_1\boldsymbol{s}_1) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times (c_2\boldsymbol{s}_2)$$

$$= c_1\left.\frac{d}{dt}\right|_{\mathcal{B}}(\boldsymbol{s}_1) + c_2\left.\frac{d}{dt}\right|_{\mathcal{B}}(\boldsymbol{s}_1) + c_1({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{s}_1) + c_2({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{s}_2)$$

$$= c_1\left[\left.\frac{d}{dt}\right|_{\mathcal{B}}(\boldsymbol{s}_1) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{s}_1\right] + c_2\left[\left.\frac{d}{dt}\right|_{\mathcal{B}}(\boldsymbol{s}_2) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{s}_2\right]$$

$$= c_1\left.\frac{d}{dt}\right|_{\mathcal{A}}(\boldsymbol{s}_1) + c_2\left.\frac{d}{dt}\right|_{\mathcal{A}}(\boldsymbol{s}_2)$$

---

8 Note the similarities of Eq. (3.41) with Eq. (**??**) in Section **??**; essentially, we have just (a) replaced the derivatives of coordinate vectors with coordinate vectors of the physical vector derivatives, and (b) added frame label $\mathcal{B}$ where appropriate.

3.7.1 What About Coordinate Vectors?

The frame derivative is an operation that is exclusive to physical vectors. However, recall that we *did* write the ***Golden Rule*** in terms of coordinate vectors as well in Eq. (3.41) in Section 3.6 (repeated below).

$$\left[\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{A}}\right]_{\mathcal{A}} = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}^{T}\left\{\left[\frac{d\left(^{\mathcal{B}}\boldsymbol{s}\right)}{dt}\bigg|_{\mathcal{B}}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}\boldsymbol{s}\right]_{\mathcal{B}}\right\} \tag{3.41}$$

While we derived this equation explicitly while developing the ***Golden Rule*** note that we could easily obtain this equation by taking Eq. (3.43) and expressing the physical vectors in the relevant frames.

> **The primary purpose of the frame derivative operator is to introduce a more generic method for deriving kinematic relationships between physical vectors.**
>
> Once these kinematic relationships have been written in terms of physical vectors, the physical vectors can be expressed in desired coordinate frames. The resulting kinematic relationships will represent the same relationships as in the physical form, but expressed in a purely mathematical (matrix algebraic) form.

3.8 Absolute Kinematics

Consider two coordinate frames, $\mathcal{A} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$ and $\mathcal{B} = \left(\hat{\boldsymbol{a}}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{c}}\right)$, that share a common origin, O. Let $\mathcal{A}$ be a stationary frame, and let $\mathcal{B}$ rotate with respect to $\mathcal{A}$ with angular velocity $^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$ and angular acceleration $^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}$. In this section, we will consider the **absolute kinematics** of a particle, P. This scenario is visualized in Fig 3.5.
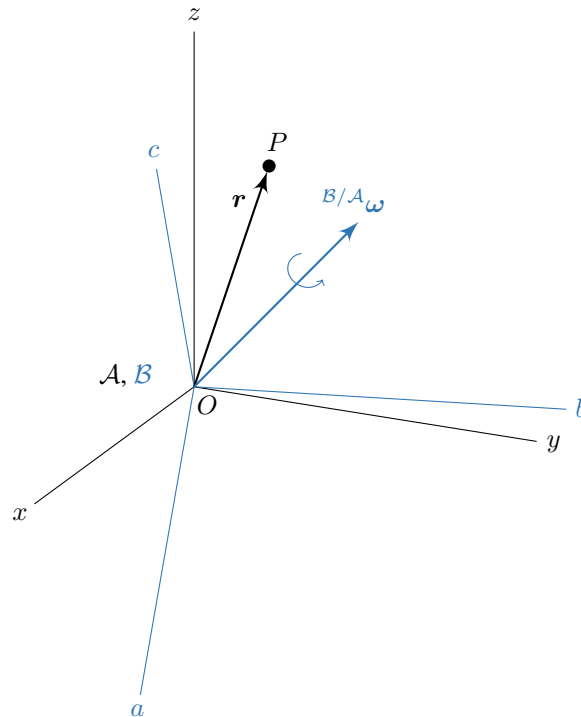
Figure 3.5: Absolute kinematics of a particle.

> **Convention 27:** Absolute kinematics definition.
>
> The **absolute kinematics** of a particle are its kinematics relative to a fixed origin.

3.8.1 Position

Consider a particle, P. The **position** (a physical vector) of the particle is denoted as r. We can express r in either coordinate frame, $\mathcal{A}$ or $\mathcal{B}$. Let's assume we know the position of P expressed in frame $\mathcal{A}$. Then its position expressed in frame $\mathcal{B}$ is

$$[r]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}[r]_{\mathcal{A}} \tag{3.44}$$

where $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$ is the rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$.

Note that the physical vector, r, is the same in both frames $\mathcal{A}$ and $\mathcal{B}$ since they share an origin. It is the *coordinates* of r that differ whether it is expressed in frame $\mathcal{A}$ or frame $\mathcal{B}$.

3.8.2 Velocity

The **velocity** (a physical vector) of a particle relative to frame $\mathcal{A}$ is defined as the frame derivative[9] of the position relative to frame $\mathcal{A}$. Thus, the velocity is a rate vector (see Section **??**), and we must specify its frame using the left superscript notation introduced in Section **??**.

$$^A v = \left.\frac{dr}{dt}\right|_{\mathcal{A}} \tag{3.45}$$

Consider the case where we have the velocity relative to frame $\mathcal{B}$, $^B v$. We know that we can find this velocity by taking the frame derivative of r with respect to frame $\mathcal{B}$.

$$^B v = \left.\frac{d}{dt}\right|_{\mathcal{B}} r = \left.\frac{dr}{dt}\right|_{\mathcal{B}}$$

By applying the definition of the frame derivative from Eq. (3.42) (Section 3.7), we can find the velocity of particle P relative to frame $\mathcal{A}$ in terms of its velocity relative to frame $\mathcal{B}$.

$$\left.\frac{d}{dt}\right|_{\mathcal{A}} r = \left.\frac{d}{dt}\right|_{\mathcal{B}} r + {}^{B/A}\boldsymbol{\omega} \times r \quad \to \quad \underbrace{\left.\frac{dr}{dt}\right|_{\mathcal{A}}}_{^A v} = \underbrace{\left.\frac{dr}{dt}\right|_{\mathcal{B}}}_{^B v} + {}^{B/A}\boldsymbol{\omega} \times r$$

$$^A v = {}^B v + {}^{B/A}\boldsymbol{\omega} \times r \tag{3.46}$$

Alternatively, we could have gotten to the last step of this simplification directly by using the physical form of the *Golden Rule* (Eq. (3.40) in Section **??**).

Equation (3.46) is written in terms of physical vectors. From a simulation perspective, we want this equation in a matrix algebra form in terms of coordinate vectors. We can do this in a single step by using the coordinate-vector form of the *Golden Rule* (Eq. (3.41)).

$$\left[\left.\frac{dr}{dt}\right|_{\mathcal{A}}\right]_{\mathcal{A}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T \left\{ \left[\left.\frac{dr}{dt}\right|_{\mathcal{B}}\right]_{\mathcal{B}} + \left[{}^{B/A}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [r]_{\mathcal{B}} \right\}$$

$$\left[{}^A v\right]_{\mathcal{A}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T \left\{ \left[{}^B v\right]_{\mathcal{B}} + \left[{}^{B/A}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [r]_{\mathcal{B}} \right\} \tag{3.47}$$

Alternatively, we could have simply taken Eq. (3.46) and expressed its physical vectors in the desired coordinate frames.

---

[9] See Section 3.7

3.8.3   Acceleration

The **acceleration** (a physical vector) of a particle relative to a coordinate frame is defined as the frame derivative of its velocity relative to that same frame. Thus, the acceleration is a rate vector (see Section **??**), and we must specify its frame using the left superscript notation introduced in Section **??**.

$$^{\mathcal{A}}\boldsymbol{a} = \frac{d\left(^{\mathcal{A}}\boldsymbol{v}\right)}{dt}\bigg|_{\mathcal{A}}$$

Since velocity itself is a derivative of position, the acceleration is the second derivative of position.

$$\boxed{^{\mathcal{A}}\boldsymbol{a} = \frac{d\left(^{\mathcal{A}}\boldsymbol{v}\right)}{dt}\bigg|_{\mathcal{A}} = \frac{d^2\boldsymbol{r}}{dt^2}\bigg|_{\mathcal{A}}} \tag{3.48}$$

We can find the acceleration of particle P relative to frame $\mathcal{A}$ by taking the frame derivative (Eq. (3.42)) relative to frame $\mathcal{A}$ of its velocity, also relative to frame $\mathcal{A}$.

$$^{\mathcal{A}}\boldsymbol{a} = \frac{d}{dt}\bigg|_{\mathcal{A}}\left(^{\mathcal{A}}\boldsymbol{v}\right)$$

Substituting Eq. (3.46),

$$^{\mathcal{A}}\boldsymbol{a} = \frac{d}{dt}\bigg|_{\mathcal{A}}\left(^{\mathcal{B}}\boldsymbol{v} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)$$

Applying the definition of the frame derivative (Eq. (3.42)),

$$^{\mathcal{A}}\boldsymbol{a} = \frac{d}{dt}\bigg|_{\mathcal{B}}\left(^{\mathcal{B}}\boldsymbol{v} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left(^{\mathcal{B}}\boldsymbol{v} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)$$

Distributing terms,

$$^{\mathcal{A}}\boldsymbol{a} = \left[\frac{d}{dt}\bigg|_{\mathcal{B}}\left(^{\mathcal{B}}\boldsymbol{v}\right) + \frac{d}{dt}\bigg|_{\mathcal{B}}\left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)\right] + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)$$

$$= \underbrace{\frac{d\left(^{\mathcal{B}}\boldsymbol{v}\right)}{dt}\bigg|_{\mathcal{B}}}_{^{\mathcal{B}}\boldsymbol{a}\text{ (Eq. (3.48))}} + \left[\frac{d}{dt}\bigg|_{\mathcal{B}}\left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right) \times \boldsymbol{r} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \frac{d}{dt}\bigg|_{\mathcal{B}}\left(\boldsymbol{r}\right)\right] + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)$$

$$= {}^{\mathcal{B}}\boldsymbol{a} + \left(\underbrace{\frac{d\left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right)}{dt}\bigg|_{\mathcal{B}}}_{^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\text{ (Eq. (3.31))}} \times \boldsymbol{r}\right) + \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \underbrace{\frac{d\boldsymbol{r}}{dt}\bigg|_{\mathcal{B}}}_{^{\mathcal{B}}\boldsymbol{v}\text{ (Eq. (3.45))}}\right) + \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}\right) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)$$

$$= {}^{\mathcal{B}}\boldsymbol{a} + \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}\right) + \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}\right) + \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}\right) + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)$$

$$\boxed{^{\mathcal{A}}\boldsymbol{a} = {}^{\mathcal{B}}\boldsymbol{a} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}\right] + \left[2\left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}\right)\right] + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)\right]} \tag{3.49}$$

Equation (3.49) is written in terms of physical vectors. From a simulation perspective, we want this equation in a matrix algebra form in terms of coordinate vectors. To do this, we can simply express the physical vectors in Eq. (3.49) in specific coordinate frames. Typically, we will have all the vectors on the right hand side expressed in frame $\mathcal{B}$; thus, we first express everything in frame $\mathcal{B}$.

$$\left[^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{B}} = \left[^{\mathcal{B}}\boldsymbol{a}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}} + 2\left(\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}}\right) + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left(\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}\right)$$

However, we will typically want the acceleration vector on the left hand side expressed in frame $\mathcal{A}$. Noting that $\left[^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}\left[^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{A}}$,

$$\mathbf{R}_{\mathcal{A}\to\mathcal{B}}\left[^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{A}} = \left[^{\mathcal{B}}\boldsymbol{a}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}} + 2\left(\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}}\right) + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left(\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}\right)$$

Left-multiplying both sides by $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T$, and noting that $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T\mathbf{R}_{\mathcal{A}\to\mathcal{B}} = \mathbf{I}_{3\times3}$,

$$\boxed{\left[{}^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{A}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T \left\{ \left[{}^{\mathcal{B}}\boldsymbol{a}\right]_{\mathcal{B}} + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}} + 2\left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[{}^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}}\right) + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}\right) \right\}}$$
$$(3.50)$$

We *could* simplify this equation slightly further by noting that $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T = \mathbf{R}_{\mathcal{B}\to\mathcal{A}}$, but in most practical situations, we will actually have knowledge of $\mathbf{R}_{\mathcal{A}\to\mathcal{B}}$.

3.8.4 Euler, Coriolis, and Centrifugal Accelerations

Recall Eq. (3.49) from the previous section that described the transformation of acceleration between frames. Each of the acceleration terms have specific names:

$$^{\mathcal{A}}\boldsymbol{a} = {}^{\mathcal{B}}\boldsymbol{a} + \underbrace{\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}\right]}_{\substack{\text{Coriolis} \\ \text{acceleration}}} + \underbrace{\left[2\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}\right)\right]}_{\substack{\text{Euler} \\ \text{acceleration}}} + \underbrace{\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)\right]}_{\substack{\text{centrifugal} \\ \text{acceleration}}}$$

In this case, since we have the angular velocity (${}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$, we are interpreting frame $\mathcal{B}$ as rotating relative to frame $\mathcal{A}$. The equation above gives us the a way to compute the acceleration in frame $\mathcal{A}$ given we know the acceleration in frame $\mathcal{B}$. We can instead get the acceleration in the rotating frame, $\mathcal{B}$, given the acceleration in the stationary frame, $\mathcal{A}$, by rearranging the above equation.

$$^{\mathcal{B}}\boldsymbol{a} = {}^{\mathcal{A}}\boldsymbol{a} - \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}\right] - \left[2\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}\right)\right] - \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)\right]$$

The acceleration terms on the right hand side are known as the **Coriolis acceleration**, **Euler acceleration**, and **centrifugal acceleration**.

$$^{\mathcal{B}}\boldsymbol{a} = {}^{\mathcal{A}}\boldsymbol{a} + \underbrace{\left\{-\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}\right]\right\}}_{\substack{\text{Coriolis} \\ \text{acceleration}}} + \underbrace{\left\{-\left[2\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}\right)\right]\right\}}_{\substack{\text{Euler} \\ \text{acceleration}}} + \underbrace{\left\{-\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)\right]\right\}}_{\substack{\text{centrifugal} \\ \text{acceleration}}}$$

These accelerations are collectively referred to as **fictitious accelerations**.

We denote fictitious accelerations using a left superscript to indicate which frame is taken to be the stationary frame and which frame is taken to be the rotating frame.

> **Convention 28:** Notation for fictitious accelerations.
>
> Consider two coordinate frames, $\mathcal{A}$ and $\mathcal{B}$. Let $\mathcal{A}$ be the stationary frame and $\mathcal{B}$ be the rotating frame, where the angular velocity and angular acceleration of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$ are ${}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$ and ${}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}$, respectively. The fictitious accelerations experienced in frame $\mathcal{B}$ due to its rotation with relative to frame $\mathcal{A}$ are denoted as
>
> $$^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{type}}$$
>
> where "type" is a placeholder for a type of fictitious acceleration (i.e. Coriolis, Euler, or centrifugal).

We thus have the following three equations for the various fictitious accelerations:

$$\boxed{{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{coriolis}} = -\,{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}} \tag{3.51}$$

$$\boxed{{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{euler}} = -2\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}\right)} \tag{3.52}$$

$$\boxed{^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{centrifugal}} = -\,^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left(^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}\right)} \tag{3.53}$$

We can then write [**ficitious_force_wikipedia**]

$$\boxed{^{\mathcal{B}}\boldsymbol{a} = {}^{\mathcal{A}}\boldsymbol{a} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{coriolis}} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{euler}} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{centrifugal}}} \tag{3.54}$$

We can also write Eqs. (3.52)–(3.54) in a coordinate vector form by expressing each vector in a certain frame. We will typically have all the vector on the right hand side of the equations expressed in the rotating frame (i.e. frame $\mathcal{B}$) in this case, so we write

$$\boxed{\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{coriolis}}\right]_{\mathcal{B}} = -\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}} \tag{3.55}$$

$$\boxed{\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{euler}}\right]_{\mathcal{B}} = -2\left(\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}}\right)} \tag{3.56}$$

$$\boxed{\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{centrifugal}}\right]_{\mathcal{B}} = -\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left(\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}\right)} \tag{3.57}$$

$$\boxed{\left[^{\mathcal{B}}\boldsymbol{a}\right]_{\mathcal{B}} = \left[^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{coriolis}}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{euler}}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{a}_{\text{centrifugal}}\right]_{\mathcal{B}}} \tag{3.58}$$

3.9 Relative Kinematics

In Section 3.8, we considered the absolute kinematics of a particle, P, relative to multiple coordinate frames. Here, we instead examine the **relative kinematics** of P relative to another particle, Q. First, we define the **relative kinematics** of particle P relative to particle Q in a single coordinate frame, $\mathcal{A} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$. This scenario is depicted in Fig. 3.6.
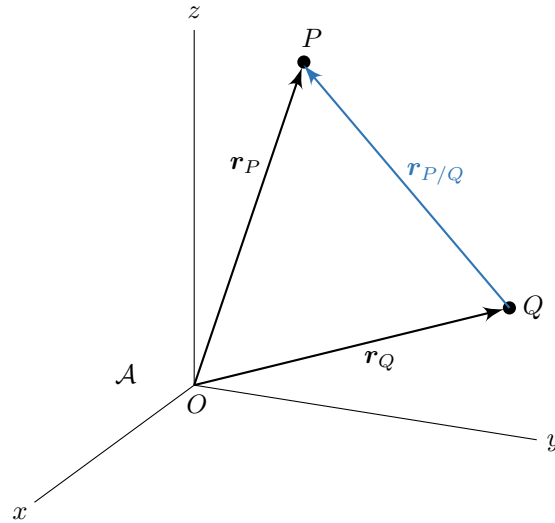
Figure 3.6: Relative kinematics.

Convention 29: Relative kinematics definition.

The **relative kinematics** of a particle are the kinematics of that particle relative to another arbitrary point that can be moving.

> Note that at the end of the day, all absolute kinematics (see Section 3.8) can also be considered as relative kinematics, where the point the kinematics are relative to is the origin of the coordinate frame(s).

Relative position

Let $\boldsymbol{r}_P$ be the position of particle P and $\boldsymbol{r}_Q$ be the position of particle Q. The **relative position** of particle P with respect to Q is defined as

$$\boxed{\boldsymbol{r}_{P/Q} = \boldsymbol{r}_P - \boldsymbol{r}_Q} \tag{3.59}$$

Relative velocity

Let ${}^{\mathcal{A}}\boldsymbol{v}_P$ be the velocity of particle P as measured in frame $\mathcal{A}$, and let ${}^{\mathcal{A}}\boldsymbol{v}_Q$ be the velocity of particle Q as measured in frame $\mathcal{A}$. The **relative velocity** of particle P with respect to Q, as measured in frame $\mathcal{A}$, is defined as

$$\boxed{{}^{\mathcal{A}}\boldsymbol{v}_{P/Q} = {}^{\mathcal{A}}\boldsymbol{v}_P - {}^{\mathcal{A}}\boldsymbol{v}_Q} \tag{3.60}$$

Relative acceleration

Let ${}^{\mathcal{A}}\boldsymbol{a}_P$ be the acceleration of particle P as measured in frame $\mathcal{A}$, and let ${}^{\mathcal{A}}\boldsymbol{a}_Q$ be the acceleration of particle Q as measured in frame $\mathcal{A}$. The **relative acceleration** of particle P with respect to Q, as measured in frame $\mathcal{A}$, is defined as [83, pp. 237–238]

$$\boxed{{}^{\mathcal{A}}\boldsymbol{a}_{P/Q} = {}^{\mathcal{A}}\boldsymbol{a}_P - {}^{\mathcal{A}}\boldsymbol{a}_Q} \tag{3.61}$$

3.9.1 Multiple Frames and Coordinate Vectors

So far in this section, we have written the relative kinematics in their most general form (as simply physical vector equations), with all vectors measured relative to the same frame, $\mathcal{A}$. However, the equations introduced thus far are completely general, and can be manipulated to get (a) rate vectors measured relative to a desired frame (can be done by substituting the relationships from Section 3.8) and (b) can be expressed in various coordinate frames. Consider Fig. 3.7, where the particles P and Q are moving relative to one another, and where coordinate frames $\mathcal{A}$ and $\mathcal{B}$ (both with origin O) are rotating with respect to one another. This is reminiscent of the scenario we discussed throughout Section 3.8, but this time we are considering P's motion relative to Q, instead of relative to the origin, O.

There are countless different ways we could express these relative kinematics; we could express each vector in different frames, and we could also consider the relative kinematics as observed from frame $\mathcal{A}$ and $\mathcal{B}$. Unfortunately, as a result, equations involving relative kinematics are typically derived on a case-by-case basis. *However*, we do derive a special case of relative kinematics for rigid body motion in Section **??**. TODO: mention next section as well
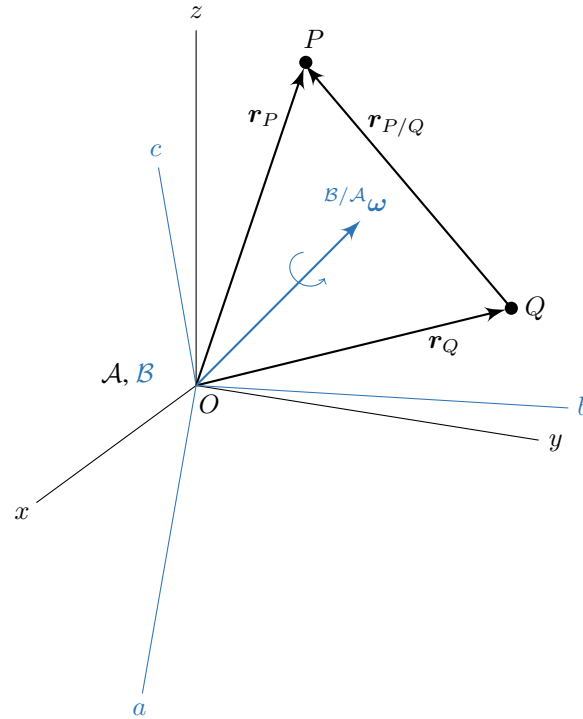
Figure 3.7: Relative kinematics with multiple coordinate frames.

3.10   Rotating vs. Moving Frames

Rotating frames

Consider two coordinate frames, $\mathcal{A} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$ and $\mathcal{B} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$, that have the same origin O, and where frame $\mathcal{B}$ rotates with respect to frame $\mathcal{A}$ with angular velocity $^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}$ and angular acceleration $^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}$. Typically, we consider one frame to be the stationary frame and one to be the "dynamic" frame. In this case, let $\mathcal{A}$ be the stationary frame. Then we refer to $\mathcal{B}$ as a rotating frame, since it only rotates with respect to frame $\mathcal{A}$.

> **Convention 30:** Rotating frame definition.
>
> A **rotating frame** is one that rotates with respect to a stationary frame, but has the same origin origin as the stationary frame for all time.

We visually depict frames $\mathcal{A}$ and $\mathcal{B}$ in Fig. 3.8. In Section 3.12, we extensively cover the transformation of kinematics between rotating and stationary frames. Note that we will first cover the transformation of kinematics between *moving* and stationary frames (Section 3.11) since rotating frames are just a subset of moving frames.

Moving frames

Until this point, we have exclusively considered kinematic transformations between rotating frames, where the coordinate frames in question shared a common origin. In the more general case, in addition to their orientations rotating about one another, the origins of these coordinate frames are also allowed to move and accelerate with respect to each
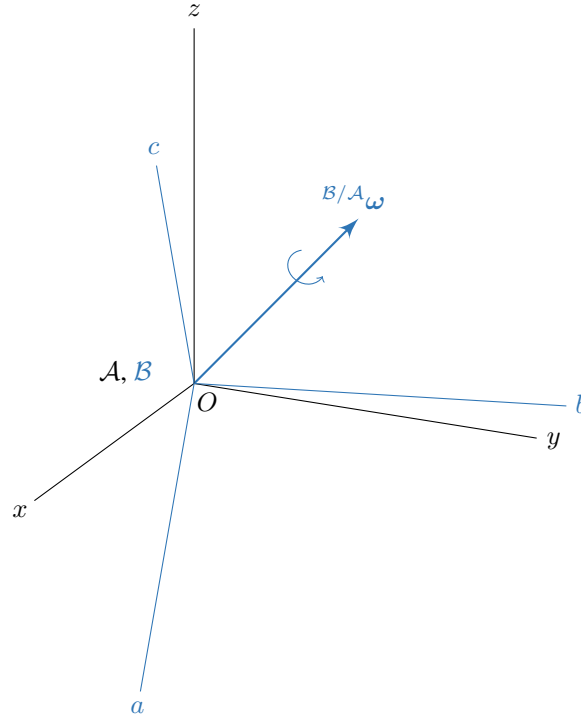
Figure 3.8: Rotating frame.

other.

> **Convention 31:** Moving frame definition.
>
> A **moving frame** is one that is both rotating *and* translating with respect to a stationary frame. In general, the origin of a rotating frame is translating relative to the origin of the stationary frame.

Consider the following two coordinate frames:

- Frame $\mathcal{A} = (\hat{x}, \hat{y}, \hat{z})$, with origin O. This frame is taken to be the stationary frame.
- Frame $\mathcal{B} = \left(\hat{a}, \hat{b}, \hat{c}\right)$, with origin O'. This frame is taken to be the moving frame.

We can consider the origins of the two frames as particles that are moving relative to one another. The origin of frame $\mathcal{B}$, O', has the following translational kinematics relative to the origin of frame $\mathcal{A}$, O:

$$r_{\mathcal{B}/\mathcal{A}} = \text{position of frame } \mathcal{B}\text{'s origin } (O') \text{ with respect to frame } \mathcal{A}\text{'s origin } (O)$$

$$^{\mathcal{A}}v_{\mathcal{B}/\mathcal{A}} = \text{velocity (as observed in frame } \mathcal{A}) \text{ of frame } \mathcal{B}\text{'s origin } (O') \text{ with respect to frame } \mathcal{A}\text{'s origin } (O)$$

$$^{\mathcal{A}}a_{\mathcal{B}/\mathcal{A}} = \text{acceleration (as observed in frame } \mathcal{A}) \text{ of frame } \mathcal{B}\text{'s origin } (O') \text{ with respect to frame } \mathcal{A}\text{'s origin } (O)$$

Additionally, like before, frame $\mathcal{B}$ has the following rotational kinematics with respect to frame $\mathcal{A}$:

$$^{\mathcal{B}/\mathcal{A}}\omega = \text{angular velocity of frame } \mathcal{B} \text{ with respect to frame } \mathcal{A}$$

$$^{\mathcal{B}/\mathcal{A}}\alpha = \text{angular acceleration of frame } \mathcal{B} \text{ with respect to frame } \mathcal{A}$$

We visually depict frames $\mathcal{A}$ and $\mathcal{B}$ in Fig. 3.9. In Section 3.11, we extensively cover the transformation of kinematics between moving and stationary frames.
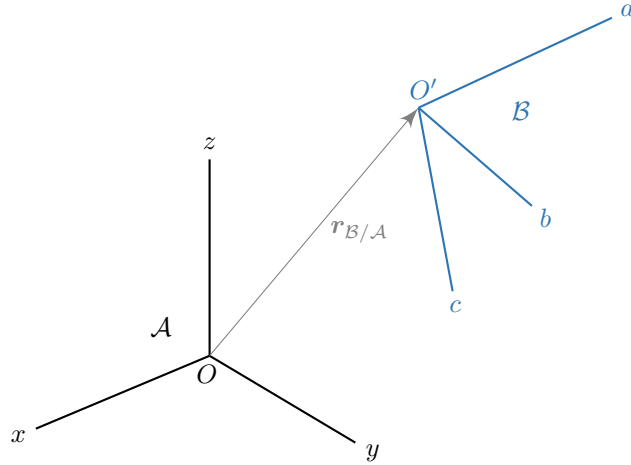
Figure 3.9: Moving frame.

3.11   Kinematic Transformations Between Moving and Stationary Frames

Recall the discussion of moving frames from the previous section. We can transform kinematics between moving and stationary frames using relative kinematics, since the origins of the stationary and moving frames can be considered as points moving relative to one another. Refer to Section 3.9 for more details on relative kinematics.

TODO: combined transformations

3.11.1   The Reverse Transformation

TODO: latex test cases for all TODO: matlab documentation for all

Consider the stationary frame, $\mathcal{A}$, and the moving frame, $\mathcal{B}$, as defined in Section 3.10. Our goal is to find the kinematics of a particle P in the stationary frame, $\mathcal{A}$, given its kinematics in the moving frame, $\mathcal{B}$. That is, we want

$$\boldsymbol{r}_{P/\mathcal{A}} = \text{position of particle } P \text{ in frame } \mathcal{A}$$
$$^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}} = \text{velocity of particle } P \text{ (as observed in frame } \mathcal{A})$$
$$^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} = \text{acceleration of particle } P \text{ (as observed in frame } \mathcal{A})$$

given the kinematics of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$ (as described in Section 3.10), as well as

$$\boldsymbol{r}_{P/\mathcal{B}} = \text{position of particle } P \text{ in frame } \mathcal{B}$$
$$^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} = \text{velocity of particle } P \text{ (as observed in frame } \mathcal{B})$$
$$^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} = \text{acceleration of particle } P \text{ (as observed in frame } \mathcal{B})$$

This scenario is visualized in Fig. 3.10. We refer to this transformation as the **reverse transformation**.

First, using Eq. (3.59) (or by simply doing a vector addition by inspection from Fig. 3.10), we can write

$$\boxed{\boldsymbol{r}_{P/\mathcal{A}} = \boldsymbol{r}_{\mathcal{B}/\mathcal{A}} + \boldsymbol{r}_{P/\mathcal{B}}} \tag{3.62}$$

Next, using Eq. (3.60), we can write

$$^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}} = {}^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}} + {}^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{B}} \tag{3.63}$$

However, given the scenario we have set up, we do not have $^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{B}}$; instead, we have $^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}$. We can resolve this by applying Eq. (3.46), which provides

$$^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{B}} = {}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}_{P/\mathcal{B}}$$

Substituting this into Eq. (3.63),

$$\boxed{^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}} = {}^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}} + {}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}_{P/\mathcal{B}}} \tag{3.64}$$
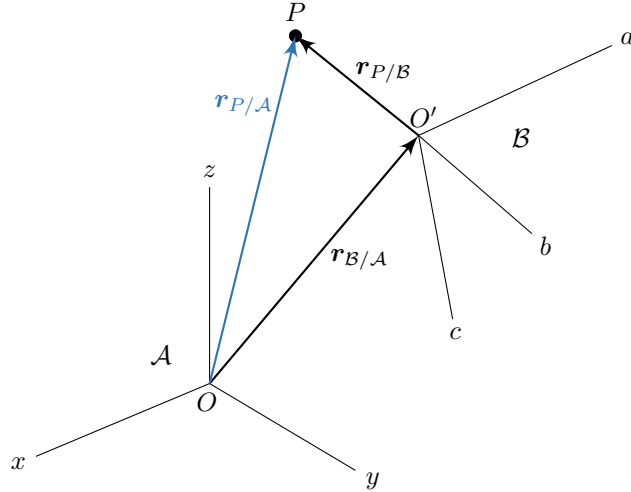
Figure 3.10: Kinematics in a stationary frame.

Finally, using Eq. (3.61), we can write

$$^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} = {}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}} + {}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{B}} \tag{3.65}$$

As was the case for the velocity, given the scenario we have set up, we do not have $^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{B}}$; instead, we have $^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}}$. We can resolve this by applying Eq. (3.49), which provides

$$^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} = {}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}_{P/\mathcal{B}} \right] + \left[2\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} \right) \right] + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}_{P/\mathcal{B}} \right) \right]$$

Substituting this into Eq. (3.65) [95, pp. 61–63], [23, pp. 28–29],

$$\boxed{^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} = {}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}} + {}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}_{P/\mathcal{B}} \right] + \left[2\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} \right) \right] + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}_{P/\mathcal{B}} \right) \right]} \tag{3.66}$$

Coordinate vector forms

We can also write these equations in a coordinate vector form by expressing all the vectors in appropriate coordinate frames and applying rotation matrices where needed. Note that Eqs. (3.64) and (3.66) are nearly identical to Eqs. (3.46) and (3.49), which we had also expressed in a coordinate vector form in Sections **??** and 3.8.3, respectively. Thus, we just express all the physical vectors with respect to the same coordinate frames here. Refer to those sections for why we chose to express certain vectors in certain coordinate frames.

$$\boxed{\left[\boldsymbol{r}_{P/\mathcal{A}} \right]_{\mathcal{A}} = \left[\boldsymbol{r}_{\mathcal{B}/\mathcal{A}} \right]_{\mathcal{A}} + \mathbf{R}_{\mathcal{A} \to \mathcal{B}}^{T} \left[\boldsymbol{r}_{P/\mathcal{B}} \right]_{\mathcal{B}}} \tag{3.67}$$

$$\boxed{\left[{}^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}} \right]_{\mathcal{A}} = \left[{}^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}} \right]_{\mathcal{A}} + \mathbf{R}_{\mathcal{A} \to \mathcal{B}}^{T} \left\{ \left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} \right]_{\mathcal{B}} + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}} \right]_{\mathcal{B}} \right\}} \tag{3.68}$$

$$\boxed{\begin{aligned} \left[{}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} \right]_{\mathcal{A}} &= \left[{}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}} \right]_{\mathcal{A}} + \mathbf{R}_{\mathcal{A} \to \mathcal{B}}^{T} \left\{ \left[{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} \right]_{\mathcal{B}} + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}} \right]_{\mathcal{B}} + 2\left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} \right]_{\mathcal{B}} \right) \right. \\ &\quad \left. + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}} \right]_{\mathcal{B}} \right) \right\} \end{aligned}} \tag{3.69}$$

Algorithms

We implement Eqs. (3.67), (3.68), and (3.69) as Algorithms 13, 14, and 15, respectively. Note that in the implementation of Eq. (3.69), we split it up into multiple steps due to its complexity.

Algorithm 13: `reverse_transform_mov_pos`

Position transformation from a moving (rotating + translating) frame to a stationary frame.

Inputs:
- $\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - position of point P with respect to frame $\mathcal{B}$ origin, expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[\boldsymbol{r}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - position of frame $\mathcal{B}$ origin with respect to frame $\mathcal{A}$ origin, expressed in frame $\mathcal{A}$

Procedure:

$$\left[\boldsymbol{r}_{P/\mathcal{A}}\right]_{\mathcal{A}} = \left[\boldsymbol{r}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} + \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^{T}\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}$$

return $\left[\boldsymbol{r}_{P/\mathcal{A}}\right]_{\mathcal{A}}$

Outputs:
- $\left[\boldsymbol{r}_{P/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - position of point P with respect to frame $\mathcal{A}$ origin, expressed in frame $\mathcal{A}$

Test Cases:
- See Appendix A.4.

Algorithm 14: `reverse_transform_mov_vel`

Velocity transformation from a moving (rotating + translating) frame to a stationary frame.

Inputs:
- $\left[^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - velocity of point P with respect to frame $\mathcal{B}$ origin, relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$
- $\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - position of point P with respect to frame $\mathcal{B}$ origin, expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ [rad/s]
- $\left[^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - velocity of frame $\mathcal{B}$ origin with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$

Procedure:

$$\left[^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}}\right]_{\mathcal{A}} = \left[^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} + \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^{T}\left\{\left[^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}\right\}$$

return $\left[^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}}\right]_{\mathcal{A}}$

Outputs:
- $\left[^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - velocity of point P with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$

Test Cases:
- See Appendix A.4.

Algorithm 15: `reverse_transform_mov_acc`

Acceleration transformation from a moving (rotating + translating) frame to a stationary frame.

Inputs:
- $\left[{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - acceleration of point P with respect to frame $\mathcal{B}$ origin, relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$
- $\left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - velocity of point P with respect to frame $\mathcal{B}$ origin, relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$
- $\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - position of point P with respect to frame $\mathcal{B}$ origin, expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular acceleration of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$
- $\left[{}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - acceleration of frame $\mathcal{B}$ origin with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$

Procedure:

1. Term 1.
$$\boldsymbol{t}_1 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}$$

2. Term 2.
$$\boldsymbol{t}_2 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \mathbf{t}_1$$

3. Term 3.
$$\boldsymbol{t}_3 = 2\left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}}\right)$$

4. Term 4.
$$\boldsymbol{t}_4 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}$$

5. Term 5.
$$\boldsymbol{t}_5 = \left[{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}}\right]_{\mathcal{B}} + \mathbf{t}_4 + \mathbf{t}_3 + \mathbf{t}_2$$

6. Acceleration of point P with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$.
$$\left[{}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}}\right]_{\mathcal{A}} = \left[{}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} + \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^{T}\mathbf{t}_5$$

7. Return the result.
$$\textbf{return } \left[{}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}}\right]_{\mathcal{A}}$$

Outputs:
- $\left[{}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} \right]_{\mathcal{A}} \in \mathbb{R}^3$ - acceleration of point P with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$

Test Cases:
- See Appendix A.4.

3.11.2 The Forward Transformation

TODO: latex test cases for all TODO: matlab documentation for all

Consider the stationary frame, $\mathcal{A}$, and the moving frame, $\mathcal{B}$, as defined in Section 3.10. Our goal is to find the kinematics of a particle P in the moving frame, $\mathcal{B}$, given its kinematics in the stationary frame, $\mathcal{A}$. That is, we want

$$\boldsymbol{r}_{P/\mathcal{B}} = \text{position of particle } P \text{ in frame } \mathcal{B}$$
$$ {}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} = \text{velocity of particle } P \text{ (as observed in frame } \mathcal{B})$$
$$ {}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} = \text{acceleration of particle } P \text{ (as observed in frame } \mathcal{B})$$

given the kinematics of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$ (as described in Section 3.10), as well as

$$\boldsymbol{r}_{P/\mathcal{A}} = \text{position of particle } P \text{ in frame } \mathcal{A}$$
$$ {}^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}} = \text{velocity of particle } P \text{ (as observed in frame } \mathcal{A})$$
$$ {}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} = \text{acceleration of particle } P \text{ (as observed in frame } \mathcal{A})$$

This scenario is visualized in Fig. 3.11. We refer to this transformation as the **forward transformation**.
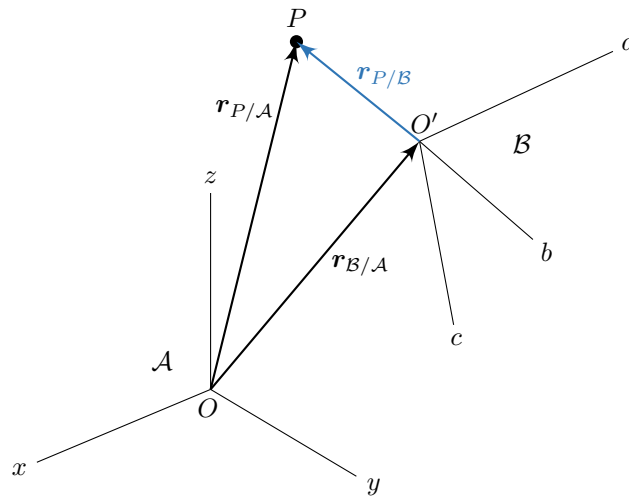
Figure 3.11: Kinematics in a moving frame.

First, we can simply rearrange Eq. (3.62) to solve for $\boldsymbol{r}_{P/\mathcal{B}}$.

$$\boxed{\boldsymbol{r}_{P/\mathcal{B}} = \boldsymbol{r}_{P/\mathcal{A}} - \boldsymbol{r}_{\mathcal{B}/\mathcal{A}}} \tag{3.70}$$

Next, we can rearrange Eq. (3.64) to solve for ${}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}$.

$$\boxed{ {}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} = -{}^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}} + {}^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}} - {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}_{P/\mathcal{B}}} \tag{3.71}$$

Note that in order to evaluate Eq. (3.64), we would first need to evaluate Eq. (3.70) for $\boldsymbol{r}_{P/\mathcal{B}}$. Finally, we can rearrange Eq. (3.66) to solve for ${}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}}$.

$$
{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} = -\,{}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}} + {}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} - \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r}_{P/\mathcal{B}}\right] - \left[2\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right)\right] - \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}_{P/\mathcal{B}}\right)\right]
$$
(3.72)

Coordinate vector forms

We can also write these equations in a coordinate vector form by expressing all the vectors in appropriate coordinate frames and applying rotation matrices where needed.

$$
\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}\left(\left[\boldsymbol{r}_{P/\mathcal{A}}\right]_{\mathcal{A}} - \left[\boldsymbol{r}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}}\right)
$$
(3.73)

$$
\left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}\left(\left[{}^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}}\right]_{\mathcal{A}} - \left[{}^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}}\right) - \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}
$$
(3.74)

$$
\left[{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}}\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}\left(\left[{}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}}\right]_{\mathcal{A}} - \left[{}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}}\right) - \left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}\right) - 2\left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}}\right)
$$
$$
- \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}\right)
$$
(3.75)

Algorithms

We implement Eqs. (3.73), (3.74), and (3.75) as Algorithms 16, 17, and 18, respectively. Note that in the implementation of Eq. (3.75), we split it up into multiple steps due to its complexity.

Algorithm 16: `forward_transform_mov_pos`

Position transformation from a stationary frame to a moving (rotating + translating) frame.

Inputs:
- $\left[\boldsymbol{r}_{P/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - position of point P with respect to frame $\mathcal{A}$ origin, expressed in frame $\mathcal{A}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[\boldsymbol{r}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - position of frame $\mathcal{B}$ origin with respect to frame $\mathcal{A}$ origin, expressed in frame $\mathcal{A}$

Procedure:

$$
\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}\left(\left[\boldsymbol{r}_{P/\mathcal{A}}\right]_{\mathcal{A}} - \left[\boldsymbol{r}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}}\right)
$$
return $\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}$

Outputs:
- $\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - position of point P with respect to frame $\mathcal{B}$ origin, expressed in frame $\mathcal{B}$

Test Cases:
- See Appendix A.4.

Algorithm 17: `forward_transform_mov_vel`

Velocity transformation from a stationary frame to a moving (rotating + translating) frame.

Inputs:
- $\left[{}^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ — velocity of point P with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$
- $\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — position of point P with respect to frame $\mathcal{B}$ origin, expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ — passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ [rad/s]
- $\left[{}^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ — velocity of frame $\mathcal{B}$ origin with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$

Procedure:

$$\left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}\left(\left[{}^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}}\right]_{\mathcal{A}} - \left[{}^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}}\right) - \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}$$

return $\left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}}$

Outputs:
- $\left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — velocity of point P with respect to frame $\mathcal{B}$ origin, relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$

Note:
- $\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}}$ can be computed using Algorithm 16.

Test Cases:
- See Appendix A.4.

Algorithm 18: `forward_transform_mov_acc`

Acceleration transformation from a stationary frame to a moving (rotating + translating) frame.

Inputs:
- $\left[{}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ — acceleration of point P with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$
- $\left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — velocity of point P with respect to frame $\mathcal{B}$ origin, relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$
- $\left[\boldsymbol{r}_{P/\mathcal{B}}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — position of point P with respect to frame $\mathcal{B}$ origin, expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ — passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ [rad/s]
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \in \mathbb{R}^3$ — angular acceleration of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ [rad/s]
- $\left[{}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}}\right]_{\mathcal{A}} \in \mathbb{R}^3$ — acceleration of frame $\mathcal{B}$ origin with respect to frame $\mathcal{A}$ origin, relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$

Procedure:

1. Term 1.

$$\mathbf{t}_1 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}} \right]_{\mathcal{B}}$$

2. Term 2.

$$\mathbf{t}_2 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \mathbf{t}_1$$

3. Term 3.

$$\mathbf{t}_3 = 2 \left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} \right]_{\mathcal{B}} \right)$$

4. Term 4.

$$\mathbf{t}_4 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \right]_{\mathcal{B}} \times \left[\boldsymbol{r}_{P/\mathcal{B}} \right]_{\mathcal{B}}$$

5. Term 5.

$$\mathbf{t}_5 = \mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}} \left(\left[{}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} \right]_{\mathcal{A}} - \left[{}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}} \right]_{\mathcal{A}} \right)$$

6. Acceleration of point P with respect to frame $\mathcal{B}$ origin, relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$.

$$\left[{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} \right]_{\mathcal{B}} = \mathbf{t}_5 - \mathbf{t}_4 - \mathbf{t}_3 - \mathbf{t}_2$$

7. Return the result.

$$\textbf{return } \left[{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} \right]_{\mathcal{B}}$$

Outputs:

- $\left[{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} \right]_{\mathcal{B}} \in \mathbb{R}^3$ - acceleration of point P with respect to frame $\mathcal{B}$ origin, relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$

Note:

- $\left[\boldsymbol{r}_{P/\mathcal{B}} \right]_{\mathcal{B}}$ can be computed using Algorithm 16.
- $\left[{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} \right]_{\mathcal{B}}$ can be computed using Algorithm 17.

Test Cases:

- See Appendix A.4.

3.11.3   Relative Kinematics in a Moving Frame

TODO

3.12   Kinematic Transformations Between Rotating and Stationary Frames

Recall the discussion of rotating frames from Section 3.10. We can transform kinematics between rotating and stationary frames using absolute kinematics, since the stationary and rotating frames share the same origin. Refer to Section

3.8 for more details on absolute kinematics.

TODO: combined transformations

3.12.1   The Reverse Transformation

TODO: latex test cases for all TODO: matlab documentation for all

Consider the stationary frame, $\mathcal{A}$, and the rotating frame, $\mathcal{B}$, as defined in Section 3.10. Our goal is to find the kinematics of a particle P in the stationary frame, $\mathcal{A}$, given its kinematics in the rotating frame, $\mathcal{B}$. That is, we want

$$\boldsymbol{r} = \text{position of particle } P \text{ in frame } \mathcal{A}$$

$$^{\mathcal{A}}\boldsymbol{v} = \text{velocity of particle } P \text{ (as observed in frame } \mathcal{A})$$

$$^{\mathcal{A}}\boldsymbol{a} = \text{acceleration of particle } P \text{ (as observed in frame } \mathcal{A})$$

given the attitude (rotational) kinematics of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$ (as described in Section 3.10), as well as

$$\boldsymbol{r} = \text{position of particle } P \text{ in frame } \mathcal{B}$$

$$^{\mathcal{B}}\boldsymbol{v} = \text{velocity of particle } P \text{ (as observed in frame } \mathcal{B})$$

$$^{\mathcal{B}}\boldsymbol{a} = \text{acceleration of particle } P \text{ (as observed in frame } \mathcal{B})$$

This scenario is visualized in Fig. 3.12. We refer to this transformation as the **reverse transformation**. Note that since
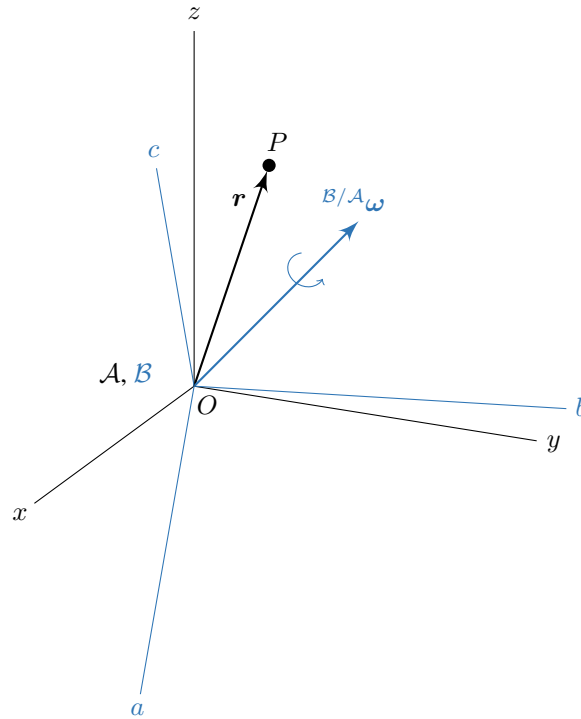
Figure 3.12: Kinematics in a stationary frame.

the two frames share the same origin, the position, $\boldsymbol{r}$, is the same in either frame. Also note that we are expressing the kinematics of the particles as absolute kinematics since the two frames have the same origin. Refer to Sections 3.8 and 3.9 for more information in absolute and relative kinematics, respectively, and to Section 3.10 for the distinction between moving and rotating frames.

As discussed in Sections 3.9 and 3.10, rotating frames are just a subset of moving frames where the origins of

the two frames remain coincident. We can thus make the following substitutions:

$$
\begin{aligned}
\boldsymbol{r}_{P/\mathcal{A}} &\to \boldsymbol{r} \\
\boldsymbol{r}_{P/\mathcal{B}} &\to \boldsymbol{r} \\
{}^{\mathcal{A}}\boldsymbol{v}_{P/\mathcal{A}} &\to {}^{\mathcal{A}}\boldsymbol{v} \\
{}^{\mathcal{B}}\boldsymbol{v}_{P/\mathcal{B}} &\to {}^{\mathcal{B}}\boldsymbol{v} \\
{}^{\mathcal{A}}\boldsymbol{a}_{P/\mathcal{A}} &\to {}^{\mathcal{A}}\boldsymbol{a} \\
{}^{\mathcal{B}}\boldsymbol{a}_{P/\mathcal{B}} &\to {}^{\mathcal{B}}\boldsymbol{a}
\end{aligned}
\tag{3.76}
$$

Additionally, the translational kinematics of the frame $\mathcal{B}$ origin relative to the frame $\mathcal{A}$ origin are just all zero, since the origins are always coincident.

$$
\begin{aligned}
\boldsymbol{r}_{\mathcal{B}/\mathcal{A}} &\to \boldsymbol{0} \\
{}^{\mathcal{A}}\boldsymbol{v}_{\mathcal{B}/\mathcal{A}} &\to \boldsymbol{0} \\
{}^{\mathcal{A}}\boldsymbol{a}_{\mathcal{B}/\mathcal{A}} &\to \boldsymbol{0}
\end{aligned}
\tag{3.77}
$$

Thus, we can just directly simplify the kinematic relationships defined in Section 3.11.1 by making the substitutions defined in Eqs. (3.76) and (3.77).

Physical vector forms

Applying the substitutions from Eqs. (3.76) and (3.77) to the kinematic relationships given by Eqs. (3.62), (3.64), and (3.66), we get

$$
\boxed{\boldsymbol{r} = \boldsymbol{r}}
\tag{3.78}
$$

$$
\boxed{{}^{\mathcal{A}}\boldsymbol{v} = {}^{\mathcal{B}}\boldsymbol{v} + {}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r}}
\tag{3.79}
$$

$$
\boxed{{}^{\mathcal{A}}\boldsymbol{a} = {}^{\mathcal{B}}\boldsymbol{a} + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \times \boldsymbol{r} \right] + \left[2\left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{v} \right) \right] + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \left({}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \times \boldsymbol{r} \right) \right]}
\tag{3.80}
$$

Note that we have directly recovered Eqs. (3.46) and (3.49) from Sections 3.8.2 and 3.8.3, respectively. Additionally, note that the position *physical* are the same in either frame since they share the same origin, as reflected by Eq. (3.78).

Coordinate vector forms

Since we recovered the physical vector forms of absolute kinematics from Section 3.8, the coordinate vector forms of Eqs. (3.78), (3.79), and (3.80) are given directly by Eqs. (3.44) (Section 3.8.1), (3.47) (Section 3.8.2), and (3.50) (Section 3.8.3), respectively.

$$
\boxed{[\boldsymbol{r}]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}[\boldsymbol{r}]_{\mathcal{A}}}
\tag{3.81}
$$

$$
\boxed{\left[{}^{\mathcal{A}}\boldsymbol{v} \right]_{\mathcal{A}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^{T} \left\{ \left[{}^{\mathcal{B}}\boldsymbol{v} \right]_{\mathcal{B}} + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}} \right\}}
\tag{3.82}
$$

$$
\boxed{\left[{}^{\mathcal{A}}\boldsymbol{a} \right]_{\mathcal{A}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^{T} \left\{ \left[{}^{\mathcal{B}}\boldsymbol{a} \right]_{\mathcal{B}} + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha} \right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}} + 2\left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \left[{}^{\mathcal{B}}\boldsymbol{v} \right]_{\mathcal{B}} \right) + \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times \left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega} \right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}} \right) \right\}}
\tag{3.83}
$$

Algorithms

We implement Eqs. (**??**), (**??**), and (**??**) as Algorithms **??**, **??**, and **??**, respectively. Note that in the implementation of Eq. (**??**), we split it up into multiple steps due to its complexity.

Algorithm 19: `reverse_transform_rot_pos`

Position transformation from a rotating frame to a stationary frame.

Inputs:
- $[r]_{\mathcal{B}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A} \to \mathcal{B}} \in \mathbb{R}^{3 \times 3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$

Procedure:

$$[r]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A} \to \mathcal{B}}[r]_{\mathcal{A}}$$

return $[r]_{\mathcal{A}}$

Outputs:
- $[r]_{\mathcal{A}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{A}$

Test Cases:
- See Appendix A.4.

Algorithm 20: `reverse_transform_rot_vel`

Velocity transformation from a rotating frame to a stationary frame.

Inputs:
- $\left[^{\mathcal{B}}v\right]_{\mathcal{B}} \in \mathbb{R}^3$ - velocity relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$
- $[r]_{\mathcal{B}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A} \to \mathcal{B}} \in \mathbb{R}^{3 \times 3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$

Procedure:

$$\left[^{\mathcal{A}}v\right]_{\mathcal{A}} = \mathbf{R}_{\mathcal{A} \to \mathcal{B}}^{T} \left\{ \left[^{\mathcal{B}}v\right]_{\mathcal{B}} + \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [r]_{\mathcal{B}} \right\}$$

return $\left[^{\mathcal{A}}v\right]_{\mathcal{A}}$

Outputs:
- $\left[^{\mathcal{A}}v\right]_{\mathcal{A}} \in \mathbb{R}^3$ - velocity relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$

Test Cases:
- See Appendix A.4.

Algorithm 21: `reverse_transform_rot_acc`

Acceleration transformation from a rotating frame to a stationary frame.

Inputs:

- $\left[{}^{\mathcal{B}}\boldsymbol{a}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - acceleration relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$
- $\left[{}^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - velocity relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$
- $[\boldsymbol{r}]_{\mathcal{B}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$
- $\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular acceleration of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$, expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$

Procedure:

1. Term 1.

$$t_1 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}$$

2. Term 2.

$$\mathbf{t}_2 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \mathbf{t}_1$$

3. Term 3.

$$\mathbf{t}_3 = 2\left(\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[{}^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}}\right)$$

4. Term 4.

$$\mathbf{t}_4 = \left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}$$

5. Term 5.

$$\mathbf{t}_5 = \left[{}^{\mathcal{B}}\boldsymbol{a}\right]_{\mathcal{B}} + \mathbf{t}_4 + \mathbf{t}_3 + \mathbf{t}_2$$

6. Acceleration relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$.

$$\left[{}^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{A}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}^T \mathbf{t}_5$$

7. Return the result.

$$\mathbf{return} \ \left[{}^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{A}}$$

Outputs:

- $\left[{}^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - acceleration relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$

Test Cases:

- See Appendix A.4.

3.12.2   The Forward Transformation

TODO: latex test cases for all TODO: matlab documentation for all

Consider the stationary frame, $\mathcal{A}$, and the rotating frame, $\mathcal{B}$, as defined in Section **??**. Our goal is to find the

kinematics of a particle P in the rotating frame, $\mathcal{B}$, given its kinematics in the stationary frame, $\mathcal{A}$. That is, we want

$$r = \text{position of particle } P \text{ in frame } \mathcal{B}$$
$$^{\mathcal{B}}v = \text{velocity of particle } P \text{ (as observed in frame } \mathcal{B})$$
$$^{\mathcal{B}}a = \text{acceleration of particle } P \text{ (as observed in frame } \mathcal{B})$$

given the attitude (rotational) kinematics of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$ (as described in Section **??**), as well as

$$r = \text{position of particle } P \text{ in frame } \mathcal{A}$$
$$^{\mathcal{A}}v = \text{velocity of particle } P \text{ (as observed in frame } \mathcal{A})$$
$$^{\mathcal{A}}a = \text{acceleration of particle } P \text{ (as observed in frame } \mathcal{A})$$

This scenario is visualized in Fig. 3.13. We refer to this transformation as the **forward transformation**. Note that
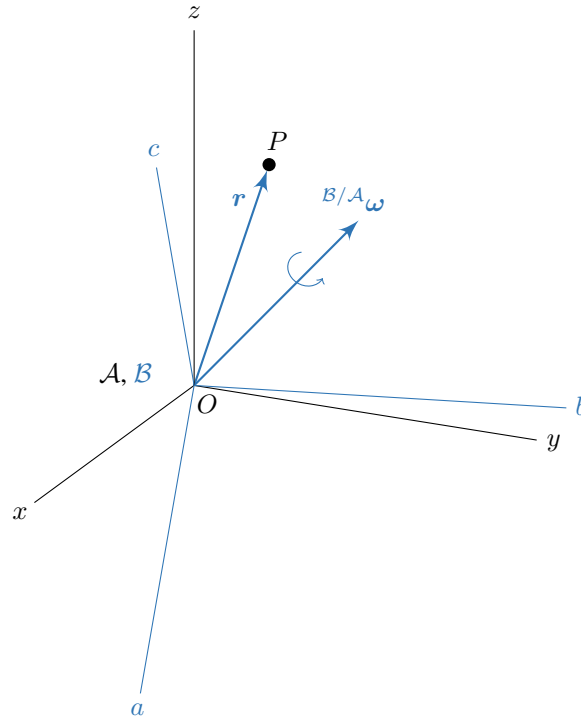
Figure 3.13: Kinematics in a rotating frame.

since the two frames share the same origin, the position, r, is the same in either frame. Also note that we are expressing the kinematics of the particles as absolute kinematics since the two frames have the same origin. Refer to Sections 3.8 and 3.9 for more information in absolute and relative kinematics, respectively, and to Section **??** for the distinction between moving and rotating frames.

Physical vector forms

Applying the substitutions from Eqs. (3.76) and (3.77) to the kinematic relationships given by Eqs. (3.70), (3.71), and (3.72), we get

$$\boxed{r = r} \tag{3.84}$$

$$\boxed{^{\mathcal{B}}v = {}^{\mathcal{A}}v - {}^{\mathcal{B}/\mathcal{A}}\omega \times r} \tag{3.85}$$

$$\boxed{^{\mathcal{B}}a = {}^{\mathcal{A}}a - \left[{}^{\mathcal{B}/\mathcal{A}}\alpha \times r \right] - \left[2\left({}^{\mathcal{B}/\mathcal{A}}\omega \times {}^{\mathcal{B}}v \right) \right] - \left[{}^{\mathcal{B}/\mathcal{A}}\omega \times \left({}^{\mathcal{B}/\mathcal{A}}\omega \times r \right) \right]} \tag{3.86}$$

Note that the position *physical* vectors are the same in either frame since they share the same origin, as reflected by Eq. (3.78).

Coordinate vector forms

We can also write these equations in a coordinate vector form by expressing all the vectors in appropriate coordinate frames and applying rotation matrices where needed. Alternatively, we could just apply the substitutions from Eqs. (3.76) and (3.77) to the kinematic relationships given by Eqs. (3.73), (3.74), and (3.75)

$$\boxed{[r]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A} \to \mathcal{B}} [r]_{\mathcal{A}}} \tag{3.87}$$

$$\boxed{\left[^{\mathcal{B}}v\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A} \to \mathcal{B}} \left[^{\mathcal{A}}v\right]_{\mathcal{A}} - \left[^{\mathcal{B}/\mathcal{A}}\omega\right]_{\mathcal{B}} \times [r]_{\mathcal{B}}} \tag{3.88}$$

$$\boxed{\left[^{\mathcal{B}}a\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A} \to \mathcal{B}} \left[^{\mathcal{A}}a\right]_{\mathcal{A}} - \left(\left[^{\mathcal{B}/\mathcal{A}}\alpha\right]_{\mathcal{B}} \times [r]_{\mathcal{B}}\right) - 2\left(\left[^{\mathcal{B}/\mathcal{A}}\omega\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}v\right]_{\mathcal{B}}\right) - \left[^{\mathcal{B}/\mathcal{A}}\omega\right]_{\mathcal{B}} \times \left(\left[^{\mathcal{B}/\mathcal{A}}\omega\right]_{\mathcal{B}} \times [r]_{\mathcal{B}}\right)}$$
$$\tag{3.89}$$

Algorithms

We implement Eqs. (**??**), (3.88), and (3.89) as Algorithms 22, 23, and 24, respectively. Note that in the implementation of Eq. (3.89), we split it up into multiple steps due to its complexity.

Algorithm 22: `forward_transform_rot_pos`

Position transformation from a stationary frame to a rotating frame.

Inputs:
- $[r]_{\mathcal{A}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{A}$
- $\mathbf{R}_{\mathcal{A} \to \mathcal{B}} \in \mathbb{R}^{3 \times 3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$

Procedure:

$$[r]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A} \to \mathcal{B}} [r]_{\mathcal{A}}$$
return $[r]_{\mathcal{B}}$

Outputs:
- $[r]_{\mathcal{B}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{B}$

Test Cases:
- See Appendix A.4.

Algorithm 23: `forward_transform_rot_vel`

Velocity transformation from a stationary frame to a moving rotating frame.

Inputs:

- $\left[^{\mathcal{A}}\boldsymbol{v}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - velocity relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$
- $[\boldsymbol{r}]_{\mathcal{B}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times 3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$,
      expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$

Procedure:

$$\left[^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}} = \mathbf{R}_{\mathcal{A}\to\mathcal{B}}\left[^{\mathcal{A}}\boldsymbol{v}\right]_{\mathcal{A}} - \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}$$

return $\left[^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}}$

Outputs:

- $\left[^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - velocity relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$

Note:

- $[\boldsymbol{r}]_{\mathcal{B}}$ can be computed using Algorithm 22.

Test Cases:

- See Appendix A.4.

Algorithm 24: `forward_transform_rot_acc`

Acceleration transformation from a stationary frame to a rotating frame.

Inputs:

- $\left[^{\mathcal{A}}\boldsymbol{a}\right]_{\mathcal{A}} \in \mathbb{R}^3$ - acceleration relative to frame $\mathcal{A}$, expressed in frame $\mathcal{A}$
- $\left[^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - velocity relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$
- $[\boldsymbol{r}]_{\mathcal{B}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{B}$
- $\mathbf{R}_{\mathcal{A}\to\mathcal{B}} \in \mathbb{R}^{3\times 3}$ - passive rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
- $\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular velocity of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$,
      expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$
- $\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \in \mathbb{R}^3$ - angular acceleration of frame $\mathcal{B}$ with respect to frame $\mathcal{A}$,
      expressed in frame $\mathcal{B}$ $[\mathrm{rad/s}]$

Procedure:

1. Term 1.

$$\mathbf{t}_1 = \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}$$

2. Term 2.

$$\mathbf{t}_2 = \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \mathbf{t}_1$$

3. Term 3.

$$\mathbf{t}_3 = 2\left(\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}} \times \left[^{\mathcal{B}}\boldsymbol{v}\right]_{\mathcal{B}}\right)$$

4. Term 4.

$$\mathbf{t}_4 = \left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}} \times [\boldsymbol{r}]_{\mathcal{B}}$$

5. Term 5.

$$\mathbf{t}_5 = \mathbf{R}_{\mathcal{A} \to \mathcal{B}} \left[{}^{\mathcal{A}}\boldsymbol{a} \right]_{\mathcal{A}}$$

6. Acceleration of point P with respect to frame $\mathcal{B}$ origin, relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$.

$$\left[{}^{\mathcal{B}}\boldsymbol{a} \right]_{\mathcal{B}} = \mathbf{t}_5 - \mathbf{t}_4 - \mathbf{t}_3 - \mathbf{t}_2$$

7. Return the result.

$$\textbf{return } \left[{}^{\mathcal{B}}\boldsymbol{a} \right]_{\mathcal{B}}$$

Outputs:
- $\left[{}^{\mathcal{B}}\boldsymbol{a} \right]_{\mathcal{B}} \in \mathbb{R}^3$ - acceleration relative to frame $\mathcal{B}$, expressed in frame $\mathcal{B}$

Note:
- $[\boldsymbol{r}]_{\mathcal{B}}$ can be computed using Algorithm 22.
- $\left[{}^{\mathcal{B}}\boldsymbol{v} \right]_{\mathcal{B}}$ can be computed using Algorithm 23.

Test Cases:
- See Appendix A.4.

3.13 Special Case: Kinematics in an Inertial Frame

Most of this chapter formulates translational kinematics in a general sense, where the coordinate frame that the kinematics are relative to is a non-inertial frame. In this section, we focus in on the specific case where all kinematics are relative to an inertial frame.

Consider an arbitrary position vector, $\boldsymbol{r}$. Taking the frame derivative relative to the inertial frame results in the inertial velocity.

$$ {}^{\mathcal{I}}\boldsymbol{v} = \frac{d}{dt}\bigg|_{\mathcal{I}} \boldsymbol{r} = \frac{d\boldsymbol{r}}{dt}\bigg|_{\mathcal{I}} $$

If we make the assumption that all kinematics are relative to an inertial frame, then we can drop all the $\mathcal{I}$ labels to de-clutter the notation.

$$ \boldsymbol{v} = \frac{d}{dt}(\boldsymbol{r}) = \frac{d\boldsymbol{r}}{dt} $$

Since we no longer label the frame on the derivative operators or on the vectors, we are free to use Newton's notation for time derivatives (see Section 3.1) to further simplify the notation.

$$ \boldsymbol{v} = \dot{\boldsymbol{r}} $$

Similarly, recall that the inertial acceleration is the frame derivative of the inertial velocity relative to the inertial frame.

$$ {}^{\mathcal{I}}\boldsymbol{a} = \frac{d}{dt}\bigg|_{\mathcal{I}} {}^{\mathcal{I}}\boldsymbol{v} = \frac{d({}^{\mathcal{I}}\boldsymbol{v})}{dt}\bigg|_{\mathcal{I}} $$

Again, with the assumption that all kinematics are relative to an inertial frame, we write

$$ \boldsymbol{a} = \frac{d}{dt}(\boldsymbol{a}) = \frac{d\boldsymbol{a}}{dt} $$

which using Newton's notation simplifies to

$$\boldsymbol{a} = \dot{\boldsymbol{v}}$$

Since $\boldsymbol{v} = \dot{\boldsymbol{r}}$, the equation above also lets us directly write the acceleration as the second derivative of position.

$$\boldsymbol{a} = \ddot{\boldsymbol{r}}$$

In summary, we have

$$\boxed{\boldsymbol{v} = \dot{\boldsymbol{r}}} \tag{3.90}$$

$$\boxed{\boldsymbol{a} = \dot{\boldsymbol{v}} = \ddot{\boldsymbol{r}}} \tag{3.91}$$

3.13.1   Magnitudes

The position ($\boldsymbol{r}$), inertial velocity ($\boldsymbol{v}$), and inertial acceleration ($\boldsymbol{a}$) magnitudes are denoted simply as

$$\boxed{\begin{aligned} r &= |\boldsymbol{r}| \\ v &= |\boldsymbol{v}| \\ a &= |\boldsymbol{a}| \end{aligned}} \tag{3.92}$$

For an arbitrary physical vector $\boldsymbol{s}$, we can write

$$\dot{s} = |\dot{\boldsymbol{s}}|$$

For inertial kinematics, this implies

$$\boxed{\begin{aligned} \dot{r} &= |\dot{\boldsymbol{r}}| \\ \dot{v} &= |\dot{\boldsymbol{v}}| \end{aligned}} \tag{3.93}$$

Since $\boldsymbol{v} = \dot{\boldsymbol{r}}$ and $\boldsymbol{a} = \dot{\boldsymbol{v}} = \ddot{\boldsymbol{r}}$, we can also write

$$\boxed{\begin{aligned} v &= \dot{r} \\ a &= \dot{v} = \ddot{r} \end{aligned}} \tag{3.94}$$

3.14   Special Case: Planar Motion in Polar Coordinates

TODO: acceleration Consider a particle, P, with absolute position $\boldsymbol{r}$ in an inertial frame, $\mathcal{I} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$. Additionally, let P be constrained to move only in the xy plane. A visual representation of this scenario is shown in Fig. 3.14. Note that the z-axis is not displayed, and is assumed to be coming out of the page. Using polar coordinates, we can parameterize the position of P in terms of the radial distance from the origin, r, and the angle between the position vector and the x-axis, θ (measured counterclockwise from the positive x-axis), as shown in Fig. 3.14.

Using polar coordinates, the motion of P can be described entirely in terms of r, θ, and their first and second time derivatives ($\dot{r}$, $\ddot{r}$, $\dot{\theta}$, and $\ddot{\theta}$). To obtain the position, velocity, and acceleration of the particle as a function of time, we introduce the rotating frame, $\mathcal{R} = (\hat{\boldsymbol{r}}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{z}})$. The **radial basis vector**, $\hat{\boldsymbol{r}}$, is always pointed in the same direction as the position vector, $\boldsymbol{r}$, i.e. the **radial direction**. The out-of-plane basis vector, $\hat{\boldsymbol{z}}$, is the same $\hat{\boldsymbol{z}}$ as for the inertial frame, $\mathcal{I} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$. The **transverse basis vector**, $\hat{\boldsymbol{\theta}}$, completes the right-handed triad, and points in the **transverse direction** (normal to the radial direction, in the plane of motion). The rotating and inertial frames are shown in Fig. 3.16. 3.14.

Our goal is to find expressions for the position, inertial velocity, and inertial acceleration. First, we note that the position is entirely along the radial direction, $\hat{\boldsymbol{r}}$. Thus,

$$\boldsymbol{r} = r\hat{\boldsymbol{r}} \tag{3.95}$$
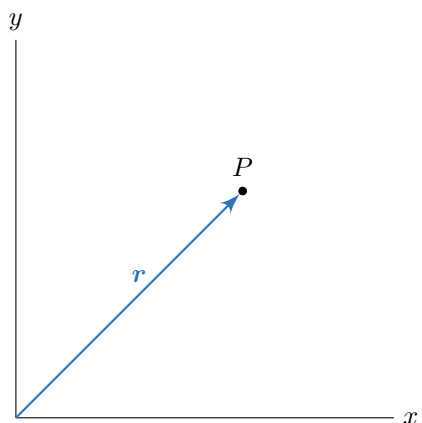
Figure 3.14: Particle in the xy plane.
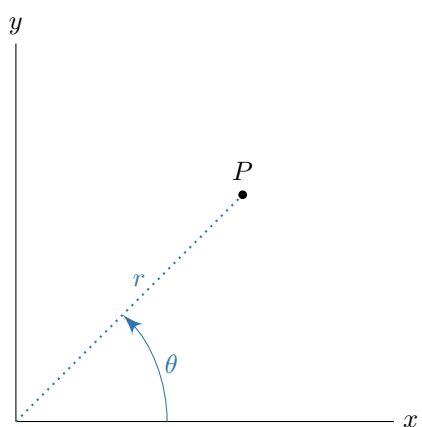
Figure 3.15: Polar coordinates.
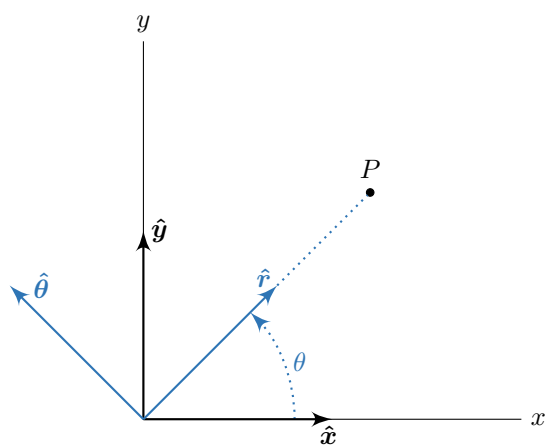
Figure 3.16: Coordinate frame definitions for polar coordinates.

To find the inertial velocity, we first note that that the magnitude of the angular velocity of the rotating frame with respect to the inertial frame is

$$^{\mathcal{R}/\mathcal{I}}\omega = \dot{\theta}$$

Since the rotation is always about the $+z$-axis (i.e. about an axis coming out of the page), and since θ is defined as

positive measured counterclockwise from the x-axis (i.e. counterclockwise rotation about the $+z$-axis), we have

$$^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} = \dot{\theta}\hat{\boldsymbol{z}} \tag{3.96}$$

Although we will not use the angular acceleration of frame $\mathcal{R}$ with respect to frame $\mathcal{I}$ in this derivation, we can define it similarly as

$$^{\mathcal{R}/\mathcal{I}}\boldsymbol{\alpha} = \ddot{\theta}\hat{\boldsymbol{z}} \tag{3.97}$$

We can find the inertial velocity, $^{\mathcal{I}}\boldsymbol{v}$, by taking the frame derivative of $\hat{\boldsymbol{r}}$ relative to the inertial frame,

$$^{\mathcal{I}}\boldsymbol{v} = \left.\frac{d}{dt}\right|_{\mathcal{I}} \boldsymbol{r}$$

Since we have the position as $\boldsymbol{r} = r\hat{\boldsymbol{r}}$, and since $\hat{\boldsymbol{r}}$ is constant relative to the rotating frame, it is convenient to apply the frame derivative operator as defined by Eq. (3.42).

$$^{\mathcal{I}}\boldsymbol{v} = \left.\frac{d}{dt}\right|_{\mathcal{R}} \boldsymbol{r} + {}^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} \times \boldsymbol{r}$$

Substituting Eqs. (3.95) and (3.96),

$$^{\mathcal{I}}\boldsymbol{v} = \left.\frac{d}{dt}\right|_{\mathcal{R}} (r\hat{\boldsymbol{r}}) + \left(\dot{\theta}\hat{\boldsymbol{z}}\right) \times (r\hat{\boldsymbol{r}}) = \dot{r}\hat{\boldsymbol{r}} + r\dot{\theta}(\hat{\boldsymbol{z}} \times \hat{\boldsymbol{r}})$$

Since $\hat{\boldsymbol{z}} \times \hat{\boldsymbol{r}} = \hat{\boldsymbol{\theta}}$,

$$^{\mathcal{I}}\boldsymbol{v} = \dot{r}\hat{\boldsymbol{r}} + r\dot{\theta}\hat{\boldsymbol{\theta}} \tag{3.98}$$

Finally, to find the inertial acceleration, we just need to take the frame derivative of the inertial velocity relative to the inertial frame. Since our vectors are written as linear combinations of the basis vectors of the rotating frame, we will use the frame derivative to rewrite the derivative relative to the inertial frame as a derivative relative to the rotating frame.

$$^{\mathcal{I}}\boldsymbol{a} = \left.\frac{d}{dt}\right|_{\mathcal{I}} {}^{\mathcal{I}}\boldsymbol{v} = \left.\frac{d}{dt}\right|_{\mathcal{R}} {}^{\mathcal{I}}\boldsymbol{v} + {}^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} \times {}^{\mathcal{I}}\boldsymbol{v}$$

Substituting Eqs. (3.96) and (3.98) [83, pp. 5–7], [37, pp. 34–35],

$$^{\mathcal{I}}\boldsymbol{a} = \left.\frac{d}{dt}\right|_{\mathcal{R}} \left(\dot{r}\hat{\boldsymbol{r}} + r\dot{\theta}\hat{\boldsymbol{\theta}}\right) + \left(\dot{\theta}\hat{\boldsymbol{z}}\right) \times \left(\dot{r}\hat{\boldsymbol{r}} + r\dot{\theta}\hat{\boldsymbol{\theta}}\right)$$

$$= \left[\ddot{r}\hat{\boldsymbol{r}} + \left(\frac{dr}{dt}\dot{\theta} + r\frac{d\dot{\theta}}{dt}\right)\hat{\boldsymbol{\theta}}\right] + \left[\dot{r}\dot{\theta}\underbrace{(\hat{\boldsymbol{z}} \times \hat{\boldsymbol{r}})}_{\hat{\boldsymbol{\theta}}} + r\dot{\theta}^2\underbrace{\left(\hat{\boldsymbol{z}} \times \hat{\boldsymbol{\theta}}\right)}_{-\hat{\boldsymbol{r}}}\right]$$

$$= \left[\ddot{r}\hat{\boldsymbol{r}} + \left(\dot{r}\dot{\theta} + r\ddot{\theta}\right)\hat{\boldsymbol{\theta}}\right] + \left[\dot{r}\dot{\theta}\hat{\boldsymbol{\theta}} - r\dot{\theta}^2\hat{\boldsymbol{r}}\right]$$

$$^{\mathcal{I}}\boldsymbol{a} = \left(\ddot{r} - r\dot{\theta}^2\right)\hat{\boldsymbol{r}} + \left(r\ddot{\theta} + 2\dot{r}\dot{\theta}\right)\hat{\boldsymbol{\theta}} \tag{3.99}$$

At this point, we have equations for $\boldsymbol{r}$, $^{\mathcal{I}}\boldsymbol{v}$, $^{\mathcal{I}}\boldsymbol{a}$, $^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega}$, and $^{\mathcal{R}/\mathcal{I}}\boldsymbol{\alpha}$ (Eqs. (3.95), (3.98), (3.99), (3.96), and (3.97), respectively). Note that all of these kinematic quantities are defined relative to the inertial frame. Therefore, we can also use Newton's notation to write $^{\mathcal{I}}\boldsymbol{v} = \dot{\boldsymbol{r}}$ and $^{\mathcal{I}}\boldsymbol{a} = \ddot{\boldsymbol{r}}$, as discussed in Section **??**. Summarizing all the equations we have derived thus far,

$$\boxed{\boldsymbol{r} = r\hat{\boldsymbol{r}}} \tag{3.100}$$

$$\boxed{^{\mathcal{I}}\boldsymbol{v} = \dot{\boldsymbol{r}} = \dot{r}\hat{\boldsymbol{r}} + r\dot{\theta}\hat{\boldsymbol{\theta}}} \tag{3.101}$$

$$\boxed{^{\mathcal{I}}\boldsymbol{a} = \ddot{\boldsymbol{r}} = \left(\ddot{r} - r\dot{\theta}^2\right)\hat{\boldsymbol{r}} + \left(r\ddot{\theta} + 2\dot{r}\dot{\theta}\right)\hat{\boldsymbol{\theta}}} \tag{3.102}$$

$$\boxed{^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} = \dot{\theta}\hat{\boldsymbol{z}}} \tag{3.103}$$

$$\boxed{^{\mathcal{R}/\mathcal{I}}\boldsymbol{\alpha} = \ddot{\theta}\hat{\boldsymbol{z}}} \tag{3.104}$$

3.14.1 Coordinate Vector Form

Equation (**??**) defines the kinematics of a particle, relative to an inertial frame, using polar coordinates. In most practical applications, we will be interested primarily in the inertial kinematics, but *expressed* in the rotating frame. Expressing Eq. (**??**) in the rotating frame,

$$[\boldsymbol{r}]_{\mathcal{R}} = \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix} \tag{3.105}$$

$$\left[{}^{\mathcal{I}}\boldsymbol{v}\right]_{\mathcal{R}} = [\dot{\boldsymbol{r}}]_{\mathcal{R}} = \begin{bmatrix} \dot{r} \\ r\dot{\theta} \\ 0 \end{bmatrix} \tag{3.106}$$

$$\left[{}^{\mathcal{I}}\boldsymbol{a}\right]_{\mathcal{R}} = [\ddot{\boldsymbol{r}}]_{\mathcal{R}} = \begin{bmatrix} \ddot{r} - r\dot{\theta}^2 \\ r\ddot{\theta} + 2\dot{r}\dot{\theta} \\ 0 \end{bmatrix} \tag{3.107}$$

$$\left[{}^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega}\right]_{\mathcal{R}} = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta} \end{bmatrix} \tag{3.108}$$

$$\left[{}^{\mathcal{R}/\mathcal{I}}\boldsymbol{\alpha}\right]_{\mathcal{R}} = \begin{bmatrix} 0 \\ 0 \\ \ddot{\theta} \end{bmatrix} \tag{3.109}$$

3.14.2 Radial and Transverse Velocity Components

When studying orbits using conic sections (see Chapter 7), it will often be useful to have the radial and transverse components of the velocity. The **radial velocity component**, v_r, is the component of the velocity in the radial direction. From Eq. (**??**), this is clearly

$$v_r = \dot{r} \tag{3.110}$$

The **transverse velocity component**, $v_\perp$, is the component of the velocity in the direction normal to the radial direction.

$$v_\perp = r\dot{\theta} \tag{3.111}$$

We will be using these definitions directly throughout Section 7.6 [23, p. 77–78]. These components are illustrated in Fig. 3.17

> We subscript the transverse component with the perpendicular symbol, $\perp$. This is because the angle will use a different symbol in almost all contexts. For example, here we use the go-to symbol for an angle, θ, but in the context of orbital mechanics we will use the symbol ν. Using the perpendicular symbol for the subscript also reminds us that this component of the velocity is normal to the radial direction.

3.14.3 Orbital Angular Velocity

Recall our discussion of spin vs. orbital angular velocity in Section 3.4.3. While the spin and orbital angular velocities are often thought of as completely different concepts, the orbital angular velocity is actually just a special case of the spin angular velocity. The orbital angular velocity defined by [6] is simply the (spin) angular velocity of a rotating frame that has its first axis defined by the position vector, and its third axis defined by the normal to the plane formed
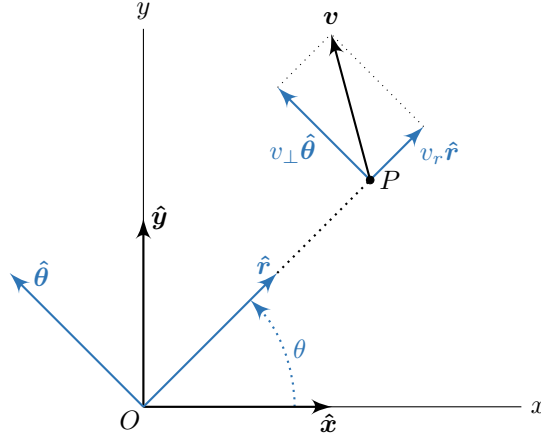
Figure 3.17: Radial and transverse velocity components.

by the position and inertial velocity vectors. Thus, the notions of "spin" and "orbital" angular velocity are intrinsically linked.

Consider the motion of a particle, P, with respect to a fixed point, O. In general, the motion of the particle P is not restricted to a single two-dimensional plane; instead, it moves in all three dimensions. However, at a single instant in time, its position and velocity vectors will lie in some plane, which we refer to as the **instantaneous plane of motion**. Essentially, at a single instant in time, the motion of the particle will be in a 2D plane. The difference between this scenario and the one we considered in the earlier parts of this section is that this plane of motion will change from one instant in time to the next.

At a single instant in time, let's construct an inertial frame $\mathcal{I}$ with its origin fixed at O and with ordered basis $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$, such that the particle's motion is entirely in the xy-plane at that instant in time. This scenario is illustrated in Fig. 3.18. At this instant in time, we can describe the motion of the particle in polar coordinates, as we have done
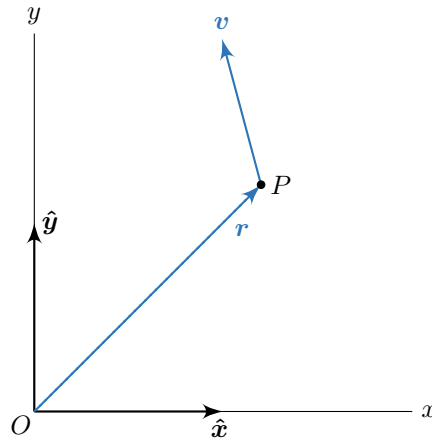
Figure 3.18: Instantaneous plane of motion.

previously in this section. The polar coordinates for this instantaneous motion are illustrated in Fig. **??**.

Earlier in Section 3.14, we have already derived the kinematics of the particle in terms of its polar coordinates. Recall that the angular velocity was given by Eq. (3.103) as

$$^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} = \dot{\theta}\hat{\boldsymbol{z}}$$

where $\mathcal{R}$ is the rotating frame with ordered basis $\left(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{z}}\right)$. In Section 3.14.2, we showed that the transverse component of the velocity, $v_\perp$, was given by

$$v_\perp = r\dot{\theta}$$
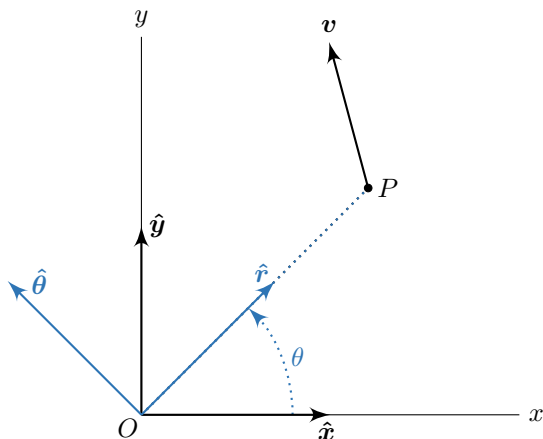
Figure 3.19: Polar coordinates of the instantaneous motion.

Solving for $\dot{\theta}$,

$$\boxed{\dot{\theta} = \frac{v_\perp}{r}}$$ (3.112)

We can simplify this further with some vector tricks. First, consider the vector diagram in Fig. **??** illustrating the radial and transverse components of v, together with an auxiliary angle, ϕ. From this diagram, we can see that
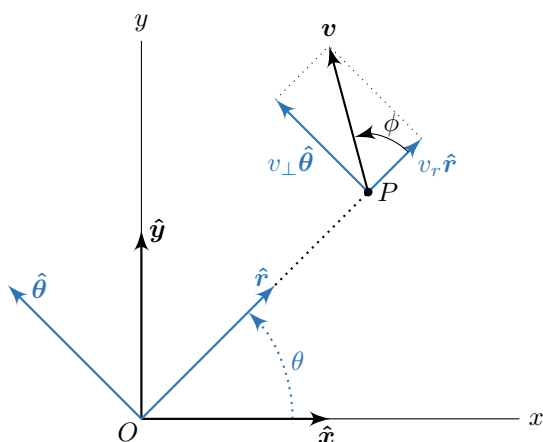
Figure 3.20: Radial and transverse velocity components with an auxiliary angle.

$$v_\perp = v \sin \phi$$

where v is the magnitude of v. Substituting this into Eq. (3.112),

$$\dot{\theta} = \frac{v \sin \phi}{r}$$

Multiplying the numerator and denominator by r,

$$\dot{\theta} = \frac{rv \sin \phi}{r^2}$$

Since ϕ is the angle between r and v, the definition of the cross product (Eq. (2.27) in Section 2.7) gives

$$|r \times v| = rv \sin \phi$$

Substituting this into the previous equation, we get

$$\dot{\theta} = \frac{|\boldsymbol{r} \times \boldsymbol{v}|}{r^2} \tag{3.113}$$

Finally, let's substitute (3.113) into Eq. (3.103).

$$^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} = \left(\frac{|\boldsymbol{r} \times \boldsymbol{v}|}{r^2} \right) \hat{\boldsymbol{z}}$$

Since $\boldsymbol{r}$ and $\boldsymbol{v}$ define the instantaneous plane of motion, and since we have defined $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{y}}$ to be coplanar with $\boldsymbol{r}$ and $\boldsymbol{v}$, we know that $\boldsymbol{r} \times \boldsymbol{v}$ is in the same direction as $\hat{\boldsymbol{z}}$ (since both are perpendicular to the instantaneous plane of motion). Thus,

$$\hat{\boldsymbol{z}} = \frac{\boldsymbol{r} \times \boldsymbol{v}}{|\boldsymbol{r} \times \boldsymbol{v}|}$$

Substituting this expression for $\hat{\boldsymbol{z}}$ into the equation for $^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega}$ above [6],

$$^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} = \left(\frac{|\boldsymbol{r} \times \boldsymbol{v}|}{r^2} \right) \left(\frac{\boldsymbol{r} \times \boldsymbol{v}}{|\boldsymbol{r} \times \boldsymbol{v}|} \right)$$

$$^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} = \frac{\boldsymbol{r} \times \boldsymbol{v}}{r^2} = \frac{\boldsymbol{r} \times \boldsymbol{v}}{\boldsymbol{r} \cdot \boldsymbol{r}} \tag{3.114}$$

Since we typically have $\boldsymbol{r}$ and $\boldsymbol{v}$ expressed in the inertial frame, $\mathcal{I}$, we can express Eq. (3.114) in the inertial frame as well.

$$\left[^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} \right]_{\mathcal{I}} = \frac{[\boldsymbol{r}]_{\mathcal{I}} \times [\boldsymbol{v}]_{\mathcal{I}}}{[\boldsymbol{r}]_{\mathcal{I}} \cdot [\boldsymbol{r}]_{\mathcal{I}}} \tag{3.115}$$

Equation (3.115) is formalized as Algorithm 25 below. Note that before, in most practical cases, we were interested in the angular velocity expressed in the rotating frame, $\left[^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} \right]_{\mathcal{R}}$. However, the primary use cases[10] of Eq. (3.115) will actually want $^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega}$ expressed in the inertial frame, and *not* in the rotating frame.

> We derived Eq. (3.115) assuming that the first two axes of $\mathcal{R}$ were instantaneously coplanar with the first two axes of $\mathcal{I}$. However, there is nothing about Eq. (3.115) that mandates that these axes be coplanar; we can use them for any fixed inertial frame $\mathcal{I}$.

Algorithm 25: `rv2omega`
Angular velocity of a rotating coordinate from position and inertial velocity.

Inputs:
- $[\boldsymbol{r}]_{\mathcal{I}} \in \mathbb{R}^3$ - position expressed in frame $\mathcal{I}$
- $[\boldsymbol{v}]_{\mathcal{I}} \in \mathbb{R}^3$ - inertial velocity expressed in frame $\mathcal{I}$

Procedure:

$$\left[^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} \right]_{\mathcal{I}} = \frac{[\boldsymbol{r}]_{\mathcal{I}} \times [\boldsymbol{v}]_{\mathcal{I}}}{[\boldsymbol{r}]_{\mathcal{I}} \cdot [\boldsymbol{r}]_{\mathcal{I}}}$$

return $\left[^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} \right]_{\mathcal{I}}$

---

[10] These uses cases are finding the true anomaly rate (Section 7.6.5) and obtaining the relative motion of a deputy satellite in the RSW frame of the chief satellite (Section TODO).

Outputs:

- $\left[^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega} \right]_{\mathcal{I}} \in \mathbb{R}^3$ - angular velocity of frame $\mathcal{R}$ with respect to frame $\mathcal{I}$, expressed in frame $\mathcal{I}$ [rad/s]

Note:

- $[\boldsymbol{r}]_{\mathcal{I}}$ and $[\boldsymbol{v}]_{\mathcal{I}}$ can be input in any units, but they *must* be consistent.
- The rotating frame, $\mathcal{R}$, is defined such that the unit vector of $[\boldsymbol{r}]_{\mathcal{I}}$ defines the 1st axis, the unit vector of $[\boldsymbol{r}]_{\mathcal{I}} \times [\boldsymbol{v}]_{\mathcal{I}}$ defines the 3rd axis, and the 2nd axis completes the right-handed triad.
- The 1st and 2nd axes of the rotating frame, $\mathcal{R}$, define the instantaneous plane of motion in which both $[\boldsymbol{r}]_{\mathcal{I}}$ and $[\boldsymbol{v}]_{\mathcal{I}}$ lie.

Test Cases:

- See Appendix A.4.

4

Attitude Kinematics

TODO: explain convention of dropping subscripts, more clearly define the body frame (put in a convention box) TODO: more on rotation matrices, make note of different sections they have already been covered in TODO: update all tikz diagrams in separate latex doc

4.1 Body Frame and World Frame

In aerospace, the **body frame** of a vehicle is a coordinate frame with its origin at the center of mass of the vehicle, and with axes aligned with the major geometric features of the vehicle. In general, the 1st axis of the body frame (x_b) aligns with the **longitudinal** (length-wise) axis of the vehicle. The third axis of the body frame (z_b) is typically chosen such that $x_b z_b$ is a plane of symmetry of the vehicle (since aerospace vehicles are typically symmetrical). The second axis of the body frame (y_b) then completes the right-handed triad (for airplanes, the y_b axis points approximately along a wing). The body frames of a rocket and an airplane are displayed in Figs. 4.1 and 4.2, respectively, below. Note that the label "cm" denotes the center of mass of the vehicle.
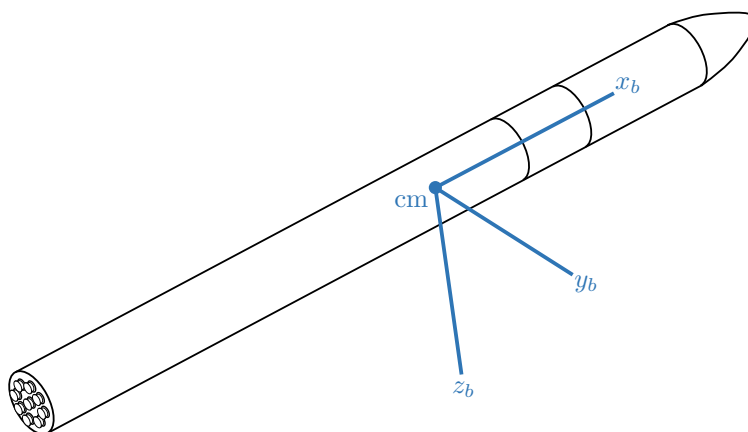
Figure 4.1: Rocket body frame.

Loosely speaking, the **attitude** of a vehicle is its orientation with respect to its environment. More rigorously, it is defined as the orientation of a vehicle's body frame with respect to the **world frame**. The world frame is essentially any other coordinate frame that we use to define a vehicle's attitude with respect to. The world and body frame are
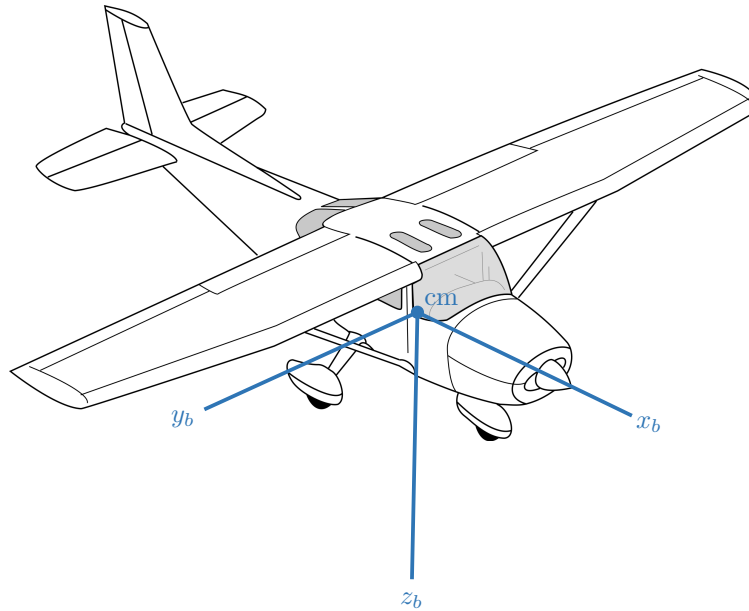
Figure 4.2: Airplane body frame.

illustrated in Fig. 4.3.
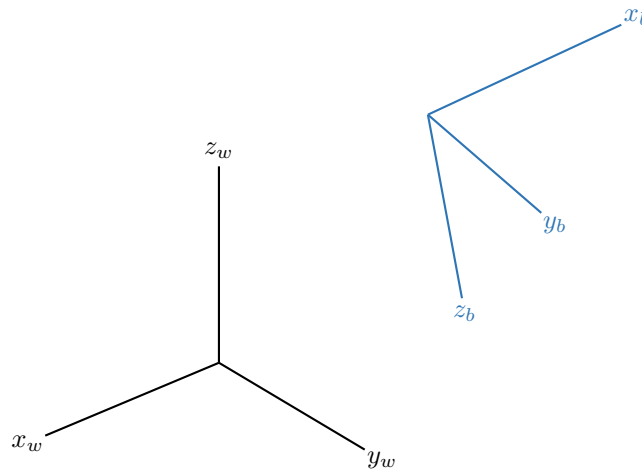
Figure 4.3: World and body frames.

4.2   Attitude, Rotation, and Direction Cosine Matrices

TODO: repeat properties, derivative, etc. To simulate both the translational and attitude dynamics of aerospace vehicles, we need a representation of its **attitude**, or orientation, with respect to its environment.

To motivate the initial development of an attitude parameterization, recall the world and body frames introduced in the previous section. A vehicle's rotational motion is most convenient to describe in the body frame, but it is (in the most general case) coupled to its translational motion, which is described in a world frame. Furthermore, many forces are also originally expressed in a world frame, so we have to transform them so we can apply them to the vehicle in the body frame. Let's consider an arbitrary force, $\boldsymbol{F}$, expressed in the world frame as $[\boldsymbol{F}]_{\text{world}}$. To express the force in

the body frame, we just need to apply the rotation matrix $\mathbf{R}_{\text{world}\to\text{body}}$.

$$[\boldsymbol{F}]_{\text{body}} = \mathbf{R}_{\text{world}\to\text{body}}[\mathbf{F}]_{\text{world}}$$

Immediately, we note that $\mathbf{R}_{\text{world}\to\text{body}}$ encodes information regarding the attitude of the vehicle.

Going into more detail, the world frame has basis $(\hat{\boldsymbol{x}}_w, \hat{\boldsymbol{y}}_w, \hat{\boldsymbol{z}}_w)$ while the body frame is defined by the basis $(\hat{\boldsymbol{x}}_b, \hat{\boldsymbol{y}}_b, \hat{\boldsymbol{z}}_b)$. At the beginning of a simulation, we will always know the basis vectors of the body frame expressed in the world frame; these serve as an initial condition for the simulation.

$$[\hat{\boldsymbol{x}}_b]_{\text{world}}, \quad [\hat{\boldsymbol{y}}_b]_{\text{world}}, \quad [\hat{\boldsymbol{z}}_b]_{\text{world}}$$

Recall from Section 2.5.2 that we can construct the rotation matrix between two frames by taking the basis vectors of one frame and expressing them in the other frame. In this case, Eq. (2.19) gives us

$$\mathbf{R}_{\text{world}\to\text{body}} = \begin{pmatrix} [\hat{\boldsymbol{x}}_b]_{\text{world}}^T \\ [\hat{\boldsymbol{y}}_b]_{\text{world}}^T \\ [\hat{\boldsymbol{z}}_b]_{\text{world}}^T \end{pmatrix}$$

As the simulation progresses, we essentially keep track of $\mathbf{R}_{\text{world}\to\text{body}}$ (usually in the form of some other attitude parameterization that can be converted to/from this form).

The **attitude matrix**, $\mathbf{A}$, is the coordinate transformation that maps vectors in the world frame to the body frame [105, p. 411]. More specifically, to match the language used in this text, the attitude matrix is the rotation matrix that takes vectors expressed in the world frame and resolves them in the body frame.

$$\mathbf{A} = \mathbf{R}_{\text{world}\to\text{body}}$$

In most practical cases, we can have multiple different world frames and multiple different body-fixed frames. Therefore, it is imperative that we also subscript the attitude matrix to specifically denote which coordinate transformation it is defining.

$$\boxed{\mathbf{A}_{\text{world}\to\text{body}} = \mathbf{R}_{\text{world}\to\text{body}}} \tag{4.1}$$

Note that the attitude matrix is also commonly referred to as the **direction cosine matrix (DCM)**, since it can be constructed using the direction cosines (see Section **??**) between each combination of basis vectors between the two frames. However, constructing an attitude matrix using direction cosines is inefficient and not used in practice, so we completely omit this definition here to avoid confusion.

4.2.1   Attitude vs. Rotation Parameterization

Through Eq. (4.1), we have just shown that the parameterization of attitude is essentially just a parameterization of rotation. Thus, we broaden the scope of the remaining sections of this chapter and discuss these parameterizations more generally as paramaterizations of *rotation*, and not just of attitude.

4.2.2   Disadvantages

Recall from Section 2.5.1 that we can chain rotation matrices as

$$\mathbf{R}_{A\to C} = \mathbf{R}_{B\to C}\mathbf{R}_{A\to B}$$

Generally, even if each individual rotation matrix is an orthogonal matrix, the product of multiple rotation matrices starts getting less and less orthogonal due to floating point errors. Due to this potential loss of orthogonality, rotation matrices can be disadvantageous to use, especially when dealing with successive rotations (recall from Section **??** that rotation matrices must be orthogonal) [35]. While we can convert the resultant rotation matrix to its nearest orthonormal representation, it is more difficult and inefficient than renormalizations needed for other rotation parameterizations (for an example, look at the quaternion normalization in Section 4.5.2).

Additionally, while we *can* easily obtain the time derivative of the rotation matrix (see Section **??**) and thus use it as a state variable when simulating an aerospace vehicle, it would (a) make the equations of attitude motion much more complicated and (b) require **nine** equations just to describe the attitude rate of change (one for each element of the rotation matrix). Furthermore, solving the differential equations numerically would almost certainly yield rotation matrices that are not orthogonal.

4.3 Euler Angles: Yaw, Pitch, and Roll

In the previous section, we defined the attitude of a vehicle using using the attitude matrix, $\mathbf{A}_{\text{world}\to\text{body}}$, which was defined as being equivalent to the rotation matrix $\mathbf{R}_{\text{world}\to\text{body}}$. Recall the 3-2-1 rotation sequence defined in Section **??**. *Any* 3D rotation can be described in terms of the 3-2-1 rotation sequence.

The **3-2-1 Euler angles** are the angles defining the 3-2-1 rotation sequence. In aerospace, we use a 3-2-1 rotation sequence to transform vectors from the world frame to the body frame. In this specific context, we refer to this rotation sequence as the **yaw-pitch-roll** sequence. As the name implies, we first perform the yaw rotation, then the pitch rotation, and finally the roll rotation. The rotation sequence can be outlined in a more detailed fashion as:

1. **1st rotation (yaw rotation):** rotation about the world z_w-axis by the **yaw angle**, ψ. This rotation transforms a vector from the $x_w y_w z_w$ coordinate system (world frame) to the intermediate coordinate system, $x'y'z'$, where $z' = z$.
2. **2nd rotation (pitch rotation):** rotation about the y'-axis by the **pitch angle**, θ. This rotation transforms a vector from the $x'y'z'$ coordinate system to another intermediate coordinate system, $x''y''z''$, where $y'' = y'$.
3. **3rd rotation (roll rotation):** rotation about the x''-axis by the **roll angle**, ϕ. This rotation transforms a vector from the $x''y''z''$ coordinate system to the $x_b y_b z_b$ coordinate system (body frame), where $x_b = x''$.

$$
\begin{array}{l}
\psi = \text{yaw angle} = \text{angle of 1st applied rotation, performed about } z_w \text{ (3rd) axis} \\
\theta = \text{pitch angle} = \text{angle of 2nd applied rotation, performed about } y' \text{ (2nd) axis} \\
\phi = \text{roll angle} = \text{angle of 3rd applied rotation, performed about } x_b = x'' \text{ (1st) axis}
\end{array} \tag{4.2}
$$

In practice, the world frame and body frame have specific names and definitions. Like we labelled the attitude matrix, we can also label the angles with the specific transformation they represent. For example:

1. $\psi_{\text{world}\to\text{body}}$ = yaw angle (world to body rotation)
2. $\theta_{\text{world}\to\text{body}}$ = pitch angle (world to body rotation)
3. $\phi_{\text{world}\to\text{body}}$ = roll angle (world to body rotation)

4.3.1 Yaw Angle

The domain of the yaw angle, ψ, is [91, p. 12]

$$
\boxed{-\pi < \psi \leq \pi} \tag{4.3}
$$

A visual depiction of the yaw angle is shown in Fig. 4.4. This simplified depiction assumes that the body z_b-axis is aligned with the world z_w-axis.

For an airplane in standard flight, a positive yaw angle corresponds to a "nose-right" attitude. *This is because the z_w and z_b axes are pointing "into the page", and yaw is positive when it defines a counterclockwise rotation about the z_w-axis.*
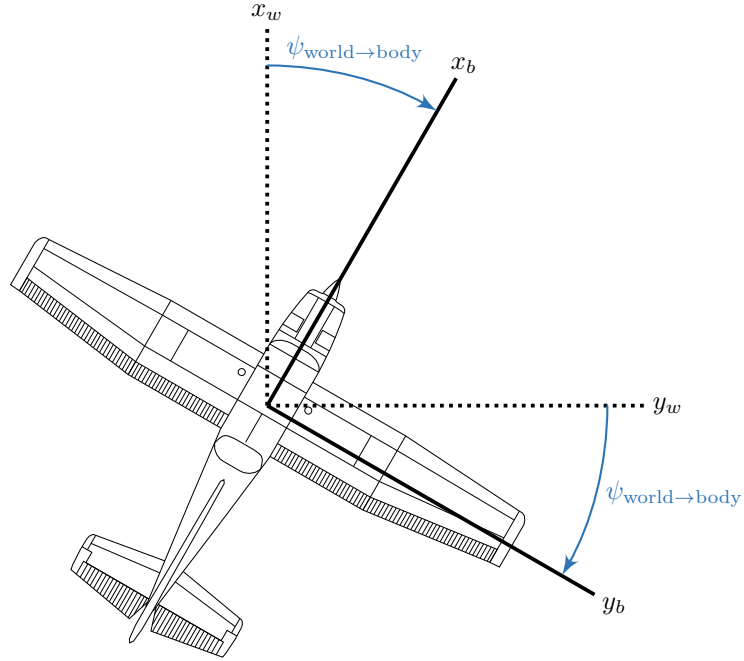
Figure 4.4: Yaw angle.

4.3.2 Pitch Angle

The domain of the pitch angle, θ, is [91, p. 12]

$$\boxed{-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}} \tag{4.4}$$

A visual depiction of the pitch angle is shown in Fig. 4.5. This simplified depiction assumes that the body y_b-axis is aligned with the world y_w-axis.

> For an airplane in standard flight, a positive pitch angle corresponds to a "nose-up" attitude. *This is because the y_w and y_b axes are pointing "into the page", and pitch is positive when it defines a counterclockwise rotation about the y_w-axis.*

4.3.3 Roll Angle

The domain of the roll angle, ϕ, is [91, p. 12]

$$\boxed{-\pi < \phi \leq \pi} \tag{4.5}$$

A visual depiction of the roll angle is shown in Fig. 4.6. This simplified depiction assumes that the body x_b-axis is aligned with the world x_w-axis.

> For an airplane in standard flight, a positive roll angle corresponds to a "bank-right" attitude. *This is because the x_w and x_b axes are pointing "out of the page", and roll is positive when it defines a counterclockwise*
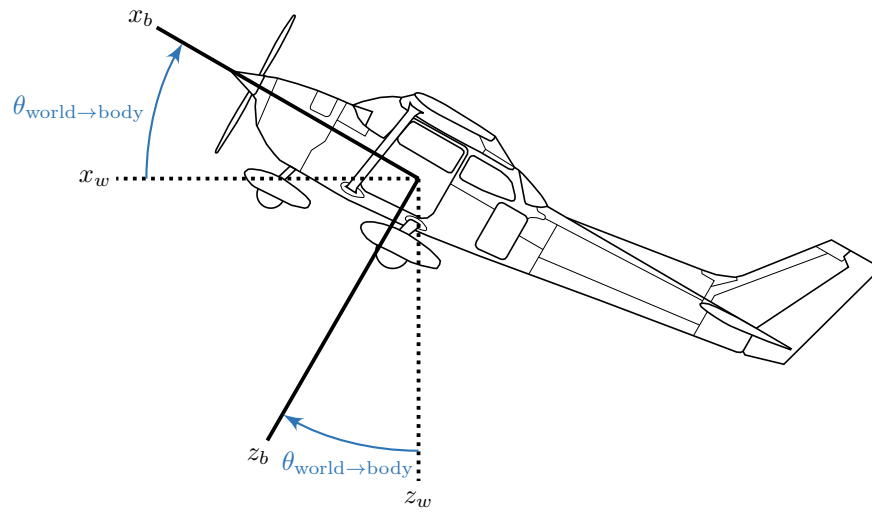
Figure 4.5: Pitch angle.
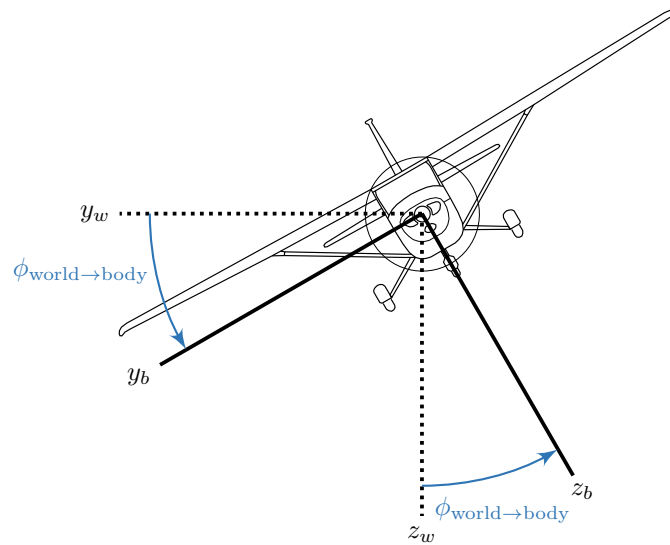
Figure 4.6: Roll angle.

rotation about the x_w-axis.

4.3.4   Relationship With Rotation Matrices

> Since we can have a rotation matrix between any two arbitrary coordinate frames, we drop the "world $\to$ body"subscripts here. However, note that in practice, it is imperative that we label the angles; otherwise, there is a complete ambiguity with regards to what transformation the angles represent.

Recall Eq. (1.44), repeated below, defining the rotation matrix for the 3-2-1 rotation sequence.

$$\mathbf{R}_{321}(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} \cos\theta_2\cos\theta_1 & \cos\theta_2\sin\theta_1 & -\sin\theta_2 \\ -\cos\theta_3\sin\theta_1 + \sin\theta_3\sin\theta_2\cos\theta_1 & \cos\theta_3\cos\theta_1 + \sin\theta_3\sin\theta_2\sin\theta_1 & \sin\theta_3\cos\theta_2 \\ \sin\theta_3\sin\theta_1 + \cos\theta_3\sin\theta_2\cos\theta_1 & -\sin\theta_3\cos\theta_1 + \cos\theta_3\sin\theta_2\sin\theta_1 & \cos\theta_3\cos\theta_2 \end{bmatrix}$$
(1.44)

Since the yaw-pitch-roll sequence is a 3-2-1 sequence, the rotation matrix can be written in terms of the yaw, pitch, and roll angles as

$$\mathbf{R} = \mathbf{R}_{321}(\psi, \theta, \phi) \tag{4.6}$$

From Eqs. (1.44) and 4.6, we get

$$\mathbf{R} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \sin\phi\cos\theta \\ \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi & \cos\phi\cos\theta \end{bmatrix} \tag{4.7}$$

However, since we already have defined Algorithm 4 for computing $\mathbf{R}_{321}(\theta_1, \theta_2, \theta_3)$, we can just wrap around that algorithm to obtain the rotation matrix from the 3-2-1 Euler angles.

Algorithm 26: `eul2mat_321`

3-2-1 Euler angles (yaw, pitch, and roll) to rotation matrix (passive rotation).

Inputs:
- $\psi \in \mathbb{R}$ - yaw angle (1st rotation, about 3rd axis) $[\text{rad}]$
- $\theta \in \mathbb{R}$ - pitch angle (2nd rotation, about 2nd axis) $[\text{rad}]$
- $\phi \in \mathbb{R}$ - roll angle (3rd rotation, about 1st axis) $[\text{rad}]$

Procedure:

    $\mathbf{R} = \texttt{rot321}(\psi, \theta, \phi)$ (Algorithm 4)

    return R

Outputs:
- $\mathbf{R} \in \mathbb{R}^{3\times3}$ - rotation matrix for the 3-2-1 (yaw-pitch-roll) rotation sequence

Test Cases:
- See Appendix A.1.1.

      We can also perform the inverse conversion; given a rotation matrix, we can obtain the yaw-pitch-roll rotation sequence that would generate that rotation matrix. Dividing R_{12} by R_{11},

$$\frac{R_{12}}{R_{11}} = \frac{\cos\theta\sin\psi}{\cos\theta\cos\psi} = \frac{\sin\psi}{\cos\psi} = \quad \to \quad \tan\psi = \frac{R_{12}}{R_{11}}$$

Since the domain of ψ spans all four quadrants, we need to solve for ψ in the correct quadrant; to do this, we can use the 4-quadrant inverse tangent.

$$\psi = \arctan2\left(R_{12}, R_{11}\right) \tag{4.8}$$

Since the domain of ψ is $(-\pi, \pi]$, and since the range of the four-quadrant inverse tangent function is $[-\pi, \pi]$, we don't have to perform any additional quadrant checks.

Next, from R_{13},

$$R_{13} = -\sin\theta \quad \rightarrow \quad \sin\theta = -R_{13} \quad \rightarrow \quad \theta = \arcsin\left(-R_{13}\right)$$

$$\theta = -\arcsin R_{13}$$

Since the domain of θ is $[-\pi/2, \pi/2]$, and since the range of the inverse sine function is $[-\pi/2, \pi/2]$, we don't have to perform any additional quadrant checks. However, in practice, it is common for $|R_{13}|$ to exceed 1 due to finite precision computer arithmetic, which is an issue since the domain of the inverse sine function is $[-1, 1]$. We can protect against this by including a check on $|A|_{13}$ [91, p. 13].

$$\theta = \begin{cases} -\arcsin R_{13}, & |R_{13}| < 1 \\ -\left(\dfrac{\pi}{2}\right)\operatorname{sgn} R_{13}, & |R_{13}| \geq 1 \end{cases} \tag{4.9}$$

Finally, dividing R_{23} by R_{33},

$$\frac{R_{23}}{R_{33}} = \frac{\sin\phi\cos\theta}{\cos\phi\cos\theta} = \frac{\sin\phi}{\cos\phi} \quad \rightarrow \quad \tan\phi = \frac{R_{23}}{R_{33}}$$

Since the domain of ϕ spans all four quadrants, we need to solve for ϕ in the correct quadrant; to do this, we can use the 4-quadrant inverse tangent.

$$\phi = \arctan2\left(R_{23}, R_{33}\right) \tag{4.10}$$

Since the domain of ϕ is $(-\pi, \pi]$, and since the range of the four-quadrant inverse tangent function is $[-\pi, \pi]$, we don't have to perform any additional quadrant checks [33], [91, p. 12].

Singularity: Pitch-Up Case

However, there remains a critical issue; the 3-2-1 Euler angles have a **singularity** when

$$\theta = \pm\frac{\pi}{2}$$

We refer to the case where $\theta = \pi/2$ as the **pitch-up case**. At this condition, we also have $R_{13} = -1$. As mentioned before, R_{13} can be ever so slightly less than -1 due to finite precision arithmetic, so we define

$$\boxed{R_{13} \leq -1 \quad \rightarrow \quad \theta = \frac{\pi}{2} \quad \rightarrow \quad \text{pitch-up case}} \tag{4.11}$$

For the pitch-up case, the rotation matrix given by Eq. (4.7) becomes

$$\mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ -\cos\phi\sin\psi + \sin\phi\cos\psi & \cos\phi\cos\psi + \sin\phi\sin\psi & 0 \\ \sin\phi\sin\psi + \cos\phi\cos\psi & -\sin\phi\cos\psi + \cos\phi\sin\psi & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & -1 \\ \sin\phi\cos\psi - \cos\phi\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\psi & 0 \\ \cos\phi\cos\psi + \sin\phi\sin\psi & -(\sin\phi\cos\psi - \cos\phi\sin\psi) & 0 \end{bmatrix}$$

Trigonometric identities give us

$$\sin\left(\phi - \psi\right) = \sin\phi\cos\psi - \cos\phi\sin\psi$$

$$\cos\left(\phi - \psi\right) = \cos\phi\cos\psi + \sin\phi\sin\psi$$

The rotation matrix can then be simplified to

$$\mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ \sin(\phi - \psi) & \cos(\phi - \psi) & 0 \\ \cos(\phi - \psi) & -\sin(\phi - \psi) & 0 \end{bmatrix}$$

If we define the auxiliary angle

$$\alpha = \phi - \psi$$

then the rotation matrix is simply

$$\mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ \sin\alpha & \cos\alpha & 0 \\ \cos\alpha & -\sin\alpha & 0 \end{bmatrix}$$

This implies that there are an infinite number of combinations of ψ and ϕ that will recover this same rotation matrix. We choose

$$\psi = 0 \tag{4.12}$$

Convention 32: Roll and yaw angles for the 3-2-1 Euler angle singularities.

When we encounter the $\theta = \pm\pi/2$ singularities while converting from another rotation parameterization to the 3-2-1 Euler angles, we manually set the yaw angle to be

$$\psi = 0$$

and then compute the roll angle, ϕ, accordingly.

The rotation matrix then becomes

$$\mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ \sin\phi & \cos\phi & 0 \\ \cos\phi & -\sin\phi & 0 \end{bmatrix}$$

From this rotation matrix, we can extract ϕ in a similar way as before.

$$\tan\phi = \frac{\sin\phi}{\cos\phi} = \frac{R_{21}}{R_{31}} \quad \rightarrow \quad \phi = \text{arctan2}\left(R_{21}, R_{31}\right) \tag{4.13}$$

Note that

Singularity: Pitch-Down Case

Now, let's handle the singularity when $\theta = -\pi/2$. We refer to the case where $\theta = -\pi/2$ as the **pitch-down case**. At this condition, we also have $R_{13} = 1$. As mentioned before, R_{13} can be ever so slightly greater than 1 due to finite precision arithmetic, so we define

$$\boxed{R_{13} \geq 1 \quad \rightarrow \quad \theta = -\frac{\pi}{2} \quad \rightarrow \quad \text{pitch-down case}} \tag{4.14}$$

For the pitch-down case, the rotation matrix given by Eq. (4.7) becomes

$$\mathbf{R} \begin{bmatrix} 0 & 0 & 1 \\ -\cos\phi\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\psi - \sin\phi\sin\psi & 0 \\ \sin\phi\sin\psi - \cos\phi\cos\psi & -\sin\phi\cos\psi - \cos\phi\sin\psi & 0 \end{bmatrix}$$

Once again, we let

$$\psi = 0 \tag{4.15}$$

resulting in

$$
\mathbf{R} = \begin{bmatrix} 0 & 0 & 1 \\ -\sin\phi & \cos\phi & 0 \\ -\cos\phi & -\sin\phi & 0 \end{bmatrix}
$$

Note that $\sin(-\phi) = -\sin\phi$ and $\cos(-\phi) = \cos\phi$. Then we have

$$
\mathbf{R} = \begin{bmatrix} 0 & 0 & 1 \\ \sin(-\phi) & \cos(-\phi) & 0 \\ -\cos(-\phi) & \sin(-\phi) & 0 \end{bmatrix}
$$

From this rotation matrix, we can extract ϕ in a similar way as before.

$$
\tan(-\phi) = \frac{\sin(-\phi)}{\cos(-\phi)} = \frac{R_{32}}{R22} \quad \rightarrow \quad -\phi = \operatorname{arctan2}(R_{32}, R_{22}) \quad \rightarrow \quad \phi = -\operatorname{arctan2}(R_{32}, R_{22}) \tag{4.16}
$$

Final Set of Equations

From Eqs. (4.8), (4.12), and (4.15), we can arrive at Eq. (4.17) defining the yaw angle. The pitch angle is still defined by Eq. (4.9), which we repeat as Eq. (4.18) below. From Eqs. (4.10), (4.13), and (4.16), we can arrive at Eq. (4.19) defining the roll angle.

$$
\psi = \begin{cases} \operatorname{arctan2}(R_{12}, R_{11}), & |R_{13}| < 1 \\ 0, & |R_{13}| \geq 1 \end{cases} \tag{4.17}
$$

$$
\theta = \begin{cases} -\arcsin R_{13}, & |R_{13}| < 1 \\ -\left(\dfrac{\pi}{2}\right)\operatorname{sgn} R_{13}, & |R_{13}| \geq 1 \end{cases} \tag{4.18}
$$

$$
\phi = \begin{cases} \operatorname{arctan2}(R_{23}, R_{33}), & |R_{13}| < 1 \\ \operatorname{arctan2}(R_{21}, R_{31}), & R_{13} \leq -1 \\ -\operatorname{arctan2}(R_{32}, R_{22}), & R_{13} \geq 1 \end{cases} \tag{4.19}
$$

Algorithm

Algorithm 27: `mat2eul_321`

Rotation matrix to 3-2-1 Euler angles (yaw, pitch, and roll) (passive rotation).

Inputs:
  • $\mathbf{R} \in \mathbb{R}^{3\times3}$ - rotation matrix for the 3-2-1 (yaw-pitch-roll) rotation sequence

Procedure:
  1. Booleans that store whether or not we are in a singular case, and if so, which one.

$$
\texttt{pitch\_up} = (R_{13} \leq -1)
$$
$$
\texttt{pitch\_down} = (R_{13} \geq 1)
$$
$$
\texttt{singular} = (\texttt{pitch\_up or pitch\_down})
$$

  2. Yaw angle [rad].

> **if** singular
> $\quad\Big|\quad \psi = 0$
> **else**
> $\quad\Big|\quad \psi = \arctan2\,(R_{12}, R_{11})$
> **end**

3. Pitch angle [rad].

> **if** singular
> $\quad\Big|\quad \theta = -\left(\dfrac{\pi}{2}\right)\operatorname{sgn} R_{13}$
> **else**
> $\quad\Big|\quad \theta = -\arcsin R_{13}$
> **end**

4. Roll angle [rad].

> **if** pitch_up
> $\quad\Big|\quad \phi = \arctan2\,(R_{21}, R_{31})$
> **else if** pitch_down
> $\quad\Big|\quad \phi = -\arctan2\,(R_{32}, R_{22})$
> **else**
> $\quad\Big|\quad \phi = \arctan2\,(R_{23}, R_{33})$
> **end**

5. Return the results.

> **return** ψ, θ, ϕ

Outputs:
- $\psi \in \mathbb{R}$ - yaw angle (1st rotation, about 3rd axis) [rad]
- $\theta \in \mathbb{R}$ - pitch angle (2nd rotation, about 2nd axis) [rad]
- $\phi \in \mathbb{R}$ - roll angle (3rd rotation, about 1st axis) [rad]

Note:
- $\psi \in (-\pi, \pi]$
- $\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$
- $\phi \in (-\pi, \pi]$
- When a singularity corresponding to the pitch angles of $\theta = \pm\pi/2$ is encountered, the yaw angle (ψ) is set to 0 and the roll angle (ϕ) is determined accordingly.

Test Cases:
- See Appendix A.1.2.

4.3.5   Gimbal Lock

Recall that in Section 4.3.4, we observed that the 3-2-1 Euler angles have singularities at $\theta = \pm\pi/2$. This singularity implied that when $\theta = \pm\pi/2$, any applied roll angle would only be seen as a rotation in yaw.

At the singularities at $\theta = \pm\pi/2$, any combination of roll and yaw rotations will result only in a rotation in yaw. Thus, we say can say that any rotation not involving pitch is *locked* to be a rotation in yaw. If we recall the pitch angle diagram in Fig. 4.5, we can note that $\pi/2$ would result in the plane would be in a vertical attitude with respect to the world frame. If we tried applying a roll to the visualization, the visualization wouldn't roll, but rather yaw. Unless we modified pitch, there would be no way to make the visualization of the plane roll if we simply constructed an rotation matrix using the Euler angles.

> For aircraft, the world frame typically has its $x_w y_w$ plane oriented parallel to the Earth's surface. For passenger planes, this singularity at $\theta = \pi/2$ does not pose much of a problem, since these planes will never fly near pitch angles of $\theta = \pi/2$. However, fighter jets and aerobatic aircraft will more often experience conditions near $\theta = \pi/2$, and rockets will almost always have a pitch angle at or near $\pi/2$ at some point in their launch trajectory.

If we construct an rotation matrix with any ψ and ϕ but with $\theta = \pi/2$, we cannot (generally[1]) recover those same values of ψ and ϕ from that same rotation matrix. As a numerical example, consider $\psi = \pi/4$, $\theta = \pi/2$, and $\phi = \pi/3$. If we construct the rotation matrix with Algorithm 26, we get

$$
\mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ 0.2588 & 0.9659 & 0 \\ 0.9659 & -0.2588 & 0.0000 \end{bmatrix}
$$

If we try recovering ϕ, θ, and ψ using Algorithm 27, we get

$$
\psi = 0
$$
$$
\theta = \frac{\pi}{2}
$$
$$
\phi = -0.2618
$$

which is not what we started out with.

Additionally, near the singularities at $\theta = \pm\pi/2$, the roll and yaw angles begin varying rapidly, and are changing infinitely fast as the attitude moves through a pitch of $\theta = \pm\pi/2$. This corresponds to extremely high Euler angle rates, which can lead to unphysical results when integrating the equations of motion using an Euler angle attitude parameterization.

In the physical world, this phenomena manifests itself as **gimbal lock**. Historically, 3-axis gyroscopes were often used for attitude determination on spacecraft. However, when any two of the three gimbals of a gyroscope are aligned, those two gimbals become locked, and the attitude with respect to the axis of those locked gimbals cannot be determined. Additionally, as the attitude moves through the singularity, the roll and yaw angles begin moving infinitely fast, to the point where they exceed the mechanical capabilities of the gimbals. In practice, it is usually the latter case that actually causes the gimbals to lock. For example, on Apollo 11, as the vehicle's attitude approached a pitch of $90°$, there was logic in place to flip one of the gimbals by $180°$ (using a torque motor) as the singularity was neared to avoid gimbal lock. However, the commanded torque was beyond the capabilities of the motor, so the system gave up and froze the gimbals instead. This required the astronauts to adjust the pitch of the spacecraft to move it away from the singularity, and then manually realign the gyroscope. In the physical case, gimbal lock can be avoided by adding a fourth gimbal to provide redundancy. This fourth gimbal is actively driven to maintain a large angle with respect to the roll and yaw axes [35].

> These concept of gimbal lock is hard to grasp (and even harder to explain). Therefore, we suggest the following additional resources:
> - http://webserver2.tecgraf.puc-rio.br/~mgattass/demo/EulerAn glesRotations-GimballLock/euler.html

---

[1] Algorithm 26 defaults to returning $\psi = 0$ if a singularity is encountered, in which case the roll angle could be correctly recovered.

- https://www.youtube.com/watch?v=zc8b2Jo7mno&t=1s
- https://en.wikipedia.org/wiki/Gimbal_lock
- https://matthew-brett.github.io/transforms3d/gimbal_lock.html

4.4 Axis-Angle Representation

TODO: express in a coordinate frame Any 3D rotation can be viewed as a single roll about a single axis. The axis of rotation (which we refer to as the **Euler axis**) is defined by a vector, $\mathbf{e} \in \mathbb{R}^3$, called the **principal rotation vector**. The angle of rotation, Φ, is referred to as the **principal angle** [95, pp. 16–19], [70]. For any 3D rotation, there are two possible axis-angle representations:

1. rotation of Φ about $\mathbf{e}$
2. rotation of $-\Phi$ about $-\mathbf{e}$

To maintain a consistent sign convention, we restrict Φ to be in the interval $[0, \pi]$, and then adjust the sign on $\mathbf{e}$ accordingly.

> **Convention 33:** Interval of the principal angle.
>
> The interval of the principal angle is
>
> $$\Phi \in [0, \pi]$$

Additionally, we assume that the principal rotation vector is a unit vector.

> **Convention 34:** Principal rotation vector magnitude.
>
> Since it is a unit vector, the principal rotation vector has a magnitude of 1.
>
> $$\|\mathbf{e}\| = \sqrt{e_1^2 + e_2^2 + e_3^2} = 1$$

To protect against numerical errors induced by floating point precision, we always normalize the principal rotation vector using

$$\mathbf{e} = \frac{\mathbf{e}}{\|\mathbf{e}\|}$$

when converting from another rotation parameterization to the axis-angle representation.

If we are rotating about the axis formed by the principal rotation vector, than the principal rotation vector must have the same coordinates before and after the rotation. Thus, for the rotation from frame A to frame B,

$$\boxed{[\mathbf{e}]_A = [\mathbf{e}]_B = \mathbf{e}} \tag{4.20}$$

We do not

4.4.1 Relationship With Rotation Matrices

Consider the rotation matrix $\mathbf{R} \in \mathbb{R}^{3 \times 3}$. Our goal is to find the principal rotation vector, $\mathbf{e} \in \mathbb{R}^3$, and the principal angle, $\Phi \in \mathbb{R}$, corresponding to the rotation matrix $\mathbf{R}$.

The principal angle, Φ, is related to the elements of the rotation matrix as

$$\cos \Phi = \frac{\operatorname{tr}(\mathbf{R}) - 1}{2} \tag{4.21}$$

where

$$\operatorname{tr}(\mathbf{R}) = R_{11} + R_{22} + R_{33} \tag{4.22}$$

is the trace of $\mathbf{R}$. This means that there are two possible solutions, Φ_1 and Φ_2, for a given rotation, where Φ_1 and Φ_2 are related as

$$\Phi_2 = \Phi_1 - 2\pi$$

This is because we can rotate about any axis rotation in two directions: clockwise and counterclockwise.

Solving for Φ using the inverse cosine function,

$$\Phi = \arccos \left[\frac{\operatorname{tr}(\mathbf{R}) - 1}{2} \right] \tag{4.23}$$

The inverse cosine function will always return a value in the interval $[0, \pi]$. This matches Convention 33. However, let's consider the case where we have constructed some rotation matrix by defining a rotation axis using a principal rotation vector, and then rotating counterclockwise about that axis by $\Phi = 3\pi/2$. If we tried to reconstruct the axis-angle representation from the rotation matrix, Eq. (4.23) would provide us with a positive angle less than π for Φ, even though the principal angle we used to construct the rotation matrix was greater than π. What is happening is the principal angle provided by Eq. (4.26) represents a rotation about a principal rotation vector antiparallel to our original principal rotation vector.

To obtain the principal rotation vector from the rotation matrix, we need to think about what happens if we try to rotate the principal rotation vector itself. If $\mathbf{e}$ is the principal rotation vector defining the axis of rotation, then applying the rotation to $\mathbf{e}$ should not change $\mathbf{e}$.

$$\mathbf{e} = \mathbf{R}\mathbf{e}$$

We can recognize this as the eigenvalue problem

$$\mathbf{R}\mathbf{e} = \lambda \mathbf{e}$$

where the eigenvalue is simply $\lambda = 1$. To solve for the principal rotation vector, $\mathbf{e}$, we simply need to find the eigenvector of $\mathbf{R}$ corresponding to the eigenvalue $\lambda = 1$. However, if we solve the eigenvalue problem numerically, we could get a result that conflicts with Φ. Eq. (4.23) will always provide Φ in the domain $[0, \pi]$, and we need to find the principal rotation vector accordingly; however, the numerical eigenvalue/eigenvector solver would have no knowledge of what quadrant we have returned Φ in. Additionally, a numerical eigenvalue/eigenvector solver can be substantially slower than simple arithmetic operations. Instead, we use an explicit solution for the eigenvector [95, pp. 16–19], [70], [10].

$$\mathbf{e} = \frac{1}{2 \sin \Phi} \begin{bmatrix} R_{23} - R_{32} \\ R_{31} - R_{13} \\ R_{12} - R_{21} \end{bmatrix} \tag{4.24}$$

Note that to avoid evaluating $\sin \Phi$, we can replace it with either

$$\sin \Phi = \sqrt{1 - \cos^2 \Phi} \tag{4.25}$$

where $\cos \Phi$ is computed using Eq. (4.21) [16, p. 35][2], or with

$$\sin \Phi = \sqrt{[3 - \operatorname{tr}(\mathbf{R})][1 + \operatorname{tr}(\mathbf{R})]}$$

---

[2] This comes from the fact that $\cos^2 \theta + \sin^2 \theta = 1$

which can be derived from substituting Eq. (4.21) into the first expression for $\sin \Phi$ above [15]. However, in practice, we will be normalizing the principal rotation vector, so it is useless to compute its coefficient.

Note that when $\Phi = 0$, there is a singularity when computing the Euler axis (see Eq. (4.24)) since $\sin \Phi = 0$. Conceptually, if no rotation occurs, it is impossible to determine the axis of rotation. We *could* just set it equal to any unit vector, but in practice, we have to take more care to ensure that we are consistent across the various rotation parameterizations. More specifically, if we choose $\mathbf{e} = (1, 0, 0)^T$, then the axis-angle to quaternion conversion covered in Section 4.5.6 will automatically yield the identity quaternion (see Section 4.5.2) corresponding to no rotation.

> **Convention 35:** Singularity handling for the axis-angle representation.
>
> The axis-angle representation has a singularity at $\Phi = 0$, corresponding to no rotation. If we are converting from some other rotation parameterization to the axis-angle representation, we default the principal rotation vector
>
> $$\mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$
>
> This allows us to automatically obtain the identity quaternion (see Section 4.5.2) corresponding to no rotation when using Algorithm 43 to convert to a quaternion.

Also, since $\cos 0 = 1$, this corresponds to the right hand side of Eq. (4.21) being equal to 1. However, there is no singularity when $\Phi = \pi$; in that case, there *is* a rotation, so we should be able to determine an axis of rotation. To find the principal rotation vector, we can simply normalize any nonzero column of $\mathbf{R} + \mathbf{I}_{3 \times 3}$ [16, pp. 32–37].

Near multiples of π, the derivative of the inverse cosine function approaches infinity, which can lead to numerical issues. To avoid this, we can instead compute the principal angle as [78]

$$\Phi = \arctan2 \left(\sin \Phi, \cos \Phi \right) \tag{4.26}$$

However, the range of the four-quadrant inverse tangent is $[-\pi, \pi]$, while we want Φ to be in $[0, \pi]$ as before. In the case that we get a negative Φ, we can just add π to it. Since the aforementioned adjustment essentially reverses the direction of rotation, we need to also reverse the direction of the principal rotation vector. Also, note that we can compute $\sin \Phi$ with Eq. (4.25) and $\cos \Phi$ with Eq. (4.21).

Algorithm 28 closely follows the algorithm[3] presented in [16, pp. 32–37, A3–A4], but uses the four-quadrant inverse tangent when computing Φ, instead of the inverse cosine function. Additionally, we do not compute the coefficient of $\mathbf{e}$, since we normalize $\mathbf{e}$ anyways. Note that the 10^{-11} tolerance on the value of $\cos \Phi$ comes from [16, p. A3]. Additionally, due to finite precision arithmetic, $\cos \Phi$ may sometimes be slightly less than -1 or slightly greater than 1, which can cause issues when solving for Φ. Thus, we manually clamp $\cos \Phi$ to have a value between -1 and 1.

> **Algorithm 28: `mat2axang`**
> Rotation matrix to axis-angle representation (passive rotation).
>
> ---
>
> **Inputs:**
>   - $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ - rotation matrix

---

[3] [16, pp. 40, A4] also presents an additional method for computing the axis-angle representation with superior numerical stability, albeit at the cost of efficiency. However, here, we use their first method, but use the numerically more stable four-quadrant inverse tangent.

Procedure:

1. Cosine of the principal angle.

$$c = \frac{R_{11} + R_{22} + R_{33} - 1}{2}$$

2. Ensure that $|c| \leq 1$.

$$c = \max\left[\min\left(c, 1\right), -1\right]$$

3. Edge case #1: $\cos \Phi = 1$.

    if $|c - 1| < 10^{-11}$

        4. Principal rotation vector.

$$\mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

        5. Principal angle $[\mathrm{rad}]$.

$$\Phi = 0$$

6. Edge case #2: $\cos \Phi = -1$.

    else if $|c + 1| < 10^{-11}$

        (a) Principal angle $[\mathrm{rad}]$.

$$\Phi = \pi$$

        (b) Auxiliary matrix.

$$\mathbf{A} = \mathbf{R} + \mathbf{I}_{3\times3}$$

        (c) Select a nonzero column of $\mathbf{A}$ and store it as $\mathbf{e}$. If no nonzero columns are found, then the principal rotation vector is just the zero vector

           $\mathbf{e} = \mathbf{0}$
           for $i = 1$ **to** 3
               if $\|\mathbf{A}_{:,i}\| > 10^{-3}$
                  $\mathbf{e} = \mathbf{A}_{:,i}$
                  break
               end
           end

7. Base case.

    else

(a) Sine of the principal angle.

$$s = \sqrt{1 - c^2}$$

(b) Principal angle.

$$\Phi = \arctan2\left(s, c\right)$$

(c) Principal rotation vector.

$$\mathbf{e} = \begin{bmatrix} R_{23} - R_{32} \\ R_{31} - R_{13} \\ R_{12} - R_{21} \end{bmatrix}$$

(d) Ensure that $\Phi \in [0, \pi]$.

if $\Phi < 0$
$\quad \Phi = \Phi + \pi$
$\quad \mathbf{e} = -\mathbf{e}$
end

end

8. Normalize the principal rotation vector.

$$\mathbf{e} = \frac{\mathbf{e}}{\|\mathbf{e}\|}$$

9. Return the results.

return $\mathbf{e}, \Phi$

Outputs:
- $\mathbf{e} \in \mathbb{R}^3$ - principal rotation vector
- $\Phi \in \mathbb{R}$ - principal angle $[\text{rad}]$

Note:
- $\|\mathbf{e}\| = 1$.
- $\Phi \in [0, \pi]$
- If $\Phi = 0$, then $\mathbf{e}$ is returned as $(1, 0, 0)^T$.

Test Cases:
- See Appendix A.1.2.

Rotation Matrix from Axis-Angle Representation

The rotation matrix can be computed from the axis-angle representation as [70], [78]

$$\mathbf{R} = \begin{bmatrix} e_1^2(1 - \cos\Phi) + \cos\Phi & e_1 e_2(1 - \cos\Phi) + e_3 \sin\Phi & e_1 e_3(1 - \cos\Phi) - e_2 \sin\Phi \\ e_2 e_1(1 - \cos\Phi) - e_3 \sin\Phi & e_2^2(1 - \cos\Phi) + \cos\Phi & e_2 e_3(1 - \cos\Phi) + e_1 \sin\Phi \\ e_3 e_1(1 - \cos\Phi) + e_2 \sin\Phi & e_3 e_2(1 - \cos\Phi) - e_1 \sin\Phi & e_3^2(1 - \cos\Phi) + \cos\Phi \end{bmatrix} \tag{4.27}$$

In a more compact matrix-algebra form, this can be written as [95, p. 43]

$$\boxed{\mathbf{R} = \mathbf{I}_{3\times3}(\cos\Phi) + (1 - \cos\Phi)\mathbf{e}\mathbf{e}^T - (\sin\Phi)\mathbf{e}^\times}$$

(4.28)

where $\mathbf{e}^\times$ is the skew-symmetric form of the column vector $\mathbf{e} \in \mathbb{R}^3$ as defined by Eq. (??).

Algorithm 29: `axang2mat`
Axis-angle representation to rotation matrix (passive rotation).

Inputs:
- $\mathbf{e} \in \mathbb{R}^3$ - principal rotation vector
- $\Phi \in \mathbb{R}$ - principal angle [rad]

Procedure:
1. Normalize the principal rotation vector.

$$\mathbf{e} = \frac{\mathbf{e}}{\|\mathbf{e}\|}$$

2. Precompute the trigonometric functions.

$$c = \cos\Phi$$
$$s = \sin\Phi$$

3. Auxiliary parameter.

$$a = 1 - c$$

4. Rotation matrix.

$$\mathbf{R} = \begin{bmatrix} e_1^2 a + c & e_1 e_2 a + e_3 s & e_1 e_3 a - e_2 s \\ e_2 e_1 a - e_3 s & e_2^2 a + c & e_2 e_3 a + e_1 s \\ e_3 e_1 a + e_2 s & e_3 e_2 a - e_1 s & e_3^2 a + c \end{bmatrix}$$

5. Return the result.

    return R

Outputs:
- $\mathbf{R} \in \mathbb{R}^{3\times3}$ - rotation matrix

Note:
- This algorithm normalizes $\mathbf{e}$.

Test Cases:
- See Appendix A.1.2.

4.4.2 Relationship With Euler Angles

Recall Eqs. (4.17), (4.18), and (4.19) defining the 3-2-1 Euler angles in terms of the elements of the rotation matrix. Also, recall Eq. (4.27) defining the elements of the rotation matrix in terms of the axis-angle representation. Using

these equations, we can obtain the 3-2-1 Euler angles corresponding to an axis-angle representation [11].

$$\psi = \begin{cases} \arctan2\left[e_1e_2(1 - \cos\Phi) + e_3\sin\Phi, e_1^2(1 - \cos\Phi) + \cos\Phi\right], & |R_{13}| < 1 \\ 0, & |R_{13}| \geq 1 \end{cases} \tag{4.29}$$

$$\theta = \begin{cases} -\arcsin R_{13}, & |R_{13}| < 1 \\ -\left(\dfrac{\pi}{2}\right)\operatorname{sgn} R_{13}, & |R_{13}| \geq 1 \end{cases} \tag{4.30}$$

$$\phi = \begin{cases} \arctan2\left[e_2e_3(1 - \cos\Phi) + e_1\sin\Phi, e_3^2(1 - \cos\Phi) + \cos\Phi\right], & |R_{13}| < 1 \\ \arctan2\left[e_2e_1(1 - \cos\Phi) - e_3\sin\Phi, e_3e_1(1 - \cos\Phi) + e_2\sin\Phi\right], & R_{13} \leq -1 \\ -\arctan2\left[e_3e_2(1 - \cos\Phi) - e_1\sin\Phi, e_2^2(1 - \cos\Phi) + \cos\Phi\right], & R_{13} \geq 1 \end{cases} \tag{4.31}$$

where

$$R_{13} = e_1e_3(1 - \cos\Phi) - e_2\sin\Phi \tag{4.32}$$

Algorithm 30 below is identical in logic to that of Algorithm 27. In principle, we are converting the axis-angle representation to a rotation matrix, and then extracting the 3-2-1 Euler angles from the rotation matrix.

Algorithm 30: `axang2eul_321`

Axis-angle representation to 3-2-1 Euler angles (yaw, pitch, and roll) (passive rotation).

Inputs:
  - $\mathbf{e} \in \mathbb{R}^3$ - principal rotation vector
  - $\Phi \in \mathbb{R}$ - principal angle [rad]

Procedure:
  1. Normalize the principal rotation vector.

$$\mathbf{e} = \frac{\mathbf{e}}{\|\mathbf{e}\|}$$

  2. Precompute the trigonometric functions.

$$c = \cos\Phi$$
$$s = \sin\Phi$$

  3. Store the elements of the principal rotation vector, $\mathbf{e} = (e_1, e_2, e_3)^T$, individually.
  4. Calculate the R_{13} element of the rotation matrix.

$$R_{13} = e_1e_3(1 - c) - e_2s$$

  5. Booleans that store whether or not we are in a singular case, and if so, which one.

$$\texttt{pitch\_up} = (R_{13} \leq -1)$$
$$\texttt{pitch\_down} = (R_{13} \geq 1)$$
$$\texttt{singular} = (\texttt{pitch\_up or pitch\_down})$$

  6. Yaw angle [rad].

if `singular`

$$\psi = 0$$

else

$$\psi = \arctan2\left[e_1 e_2(1 - c) + e_3 s, e_1^2(1 - c) + c\right]$$

end

7. Pitch angle [rad].

if `singular`

$$\theta = -\left(\frac{\pi}{2}\right)\operatorname{sgn}R_{13}$$

else

$$\theta = -\arcsin R_{13}$$

end

8. Roll angle [rad].

if `pitch_up`

$$\phi = \arctan2\left[e_2 e_1(1 - c) - e_3 s, e_3 e_1(1 - c) + e_2 s\right]$$

else if `pitch_down`

$$\phi = -\arctan2\left[e_3 e_2(1 - c) - e_1 s, e_2^2(1 - c) + c\right]$$

else

$$\phi = \arctan2\left[e_2 e_3(1 - c) + e_1 s, e_3^2(1 - c) + c\right]$$

end

9. Return the results.

return ψ, θ, ϕ

Outputs:
- $\psi \in \mathbb{R}$ - yaw angle (1st rotation, about 3rd axis) [rad]
- $\theta \in \mathbb{R}$ - pitch angle (2nd rotation, about 2nd axis) [rad]
- $\phi \in \mathbb{R}$ - roll angle (3rd rotation, about 1st axis) [rad]

Note:
- This algorithm normalizes **e**.
- $\psi \in (-\pi, \pi]$
- $\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$
- $\phi \in (-\pi, \pi]$
- When a singularity corresponding to the pitch angles of $\theta = \pm\pi/2$ is encountered, the yaw angle (ψ) is set to 0 and the roll angle (ϕ) is determined accordingly.

Test Cases:
- See Appendix A.1.2.

As we go through all these rotation parameterizations and the conversions between them, note that we write all conversions in an explicit form, i.e. not depending on an intermediate conversion. The one exception is the conversion from 3-2-1 Euler angles to the axis-angle representation. For this conversion, we actually skip ahead to the material covered in Section 4.5.5 and first convert the 3-2-1 Euler angles to a quaternion using Algorithm 42. We then obtain the axis-angle representation from the quaternion using Algorithm 44.

Algorithm 31: `eul2axang_321`

3-2-1 Euler angles (yaw, pitch, and roll) to axis-angle representation (passive rotation).

Inputs:
- $\psi \in \mathbb{R}$ - yaw angle (1st rotation, about 3rd axis) [rad]
- $\theta \in \mathbb{R}$ - pitch angle (2nd rotation, about 2nd axis) [rad]
- $\phi \in \mathbb{R}$ - roll angle (3rd rotation, about 1st axis) [rad]

Procedure:
1. Unit quaternion (Algorithm 42).

$$\mathbf{q} = \texttt{eul2quat\_321}(\psi, \theta, \phi)$$

2. Principal rotation vector and principal angle [rad] (Algorithm 44).

$$[\mathbf{e}, \Phi] = \texttt{quat2axang}(\mathbf{q})$$

3. Return the results.

$$\textbf{return } \psi, \theta, \phi$$

Outputs:
- $\mathbf{e} \in \mathbb{R}^3$ - principal rotation vector
- $\Phi \in \mathbb{R}$ - principal angle [rad]

Note:
- $\|\mathbf{e}\| = 1$.
- $\Phi \in [0, \pi]$
- If $\Phi = 0$, then $\mathbf{e}$ is returned as $(1, 0, 0)^T$.

Test Cases:
- See Appendix A.1.2.

4.4.3 Disadvantages

The axis-angle representation is not unique, since a rotation of $-\Phi$ about $-\mathbf{e}$ is the same as a rotation of Φ about $\mathbf{e}$ [10]. Additionally, it is not possible to determine a principal rotation vector for a principal angle of 0. Finally, there is an ambiguity in the direction of the principal rotation vector at $\Phi = \pi$.

4.5 Quaternions

A **quaternion** is a four-dimensional vector that can be used to parameterize a rotation. It consists of a scalar part, q_0, and a vector part, $\mathbf{q}_v = (q_1, q_2, q_3)^T$.

$$\mathbf{q} = \begin{bmatrix} q_0 \\ \mathbf{q}_v \end{bmatrix} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{4.33}$$

> **Convention 36:** Scalar-first convention.
>
> Throughout this text, we use the **scalar-first convention**, where the scalar part of the quaternion is the first element of the quaternion, as shown in Eq. (4.33). This is the most widely-used convention in aerospace.
>
> In some cases, the scalar-*last* convention is used, where the quaternion is defined as $\mathbf{q} = (q_1, q_2, q_3, q_4)^T$, where q_4 is the scalar part and where $\mathbf{q}_v = (q_1, q_2, q_3)^T$ is still the vector part.

In general, quaternions do not have to have unit magnitude. However, when used for rotation parameterization, we deal specifically with **unit quaternions**. Unit quaternions are covered in more detail in Section 4.5.2.

> We will be referring to quaternions and unit quaternions interchangeably, but typically, when we refer to a quaternion in this text, we are referring to a *unit* quaternion. However, the algorithms that take a quaternion and convert it to an alternate representation do not make the assumption that we are dealing with unit quaternions; they will normalize the quaternions by default.

Like the axis-angle representation, unit quaternions have a choice of sign[4]; that is, for the purposes of representing rotations, $\mathbf{q}$ is equivalent to $-\mathbf{q}$.

$$\boxed{\mathbf{q} \equiv -\mathbf{q}} \tag{4.34}$$

Due to this freedom in picking a sign, we have to select a convention. We choose for the scalar part to be positive.

> **Convention 37:** Quaternion sign convention.
>
> The scalar part of a quaternion, q_0, should be positive.

See Section 4.5.6 for more information on the relationship between unit quaternions and the axis-angle representation.

> In practice, it is essential to subscript the unit quaternions with what rotation they represent. For example, if we are transforming the coordinates of a vector from frame A to frame B, and are representing that rotation using a unit quaternion, then we would label that unit quaternion as $\mathbf{q}_{A \to B}$.

4.5.1 Treatment as a Column Vector

The mathematical details of quaternions require a greater amount of mathematical maturity than rotation matrices and Euler angles. These details are generally covered in greater detail in most other texts, but they do not often make it clear how to *use* quaternions. Here, we focus simply on the implementation side.

The main thing we *do* want to note is regarding quaternions is that the set of quaternions forms its own vector space, $\mathbb{H}$; quaternions are ***not*** members of the vector space $\mathbb{R}^4$. In a more traditional mathematical sense, quaternions are typically defined as

$$\mathbf{q} = q_0 + q_1 \mathbf{i} + q_2 \mathbf{j} + q_3 \mathbf{k}$$

---

[4] It is incorrect to refer to this as a sign ambiguity, since regardless of the sign you choose, there is not ambiguity as to which rotation the unit quaternion represents

where

$$q_0, q_1, q_2, q_3 \in \mathbb{R}$$

and where $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$ are symbols that can be interpreted as unit vectors point along three spatial axes. Note that this is very similar to how we can write a complex number, $z \in \mathbb{C}$, as

$$z = a + bi$$

where $a, b \in \mathbb{R}$ and where i is the imaginary unit. Just like how complex numbers are often analogous to vectors in $\mathbb{R}^2$ (i.e. they are plotted as vectors in the imaginary plane), we can often treat a quaternion like it is a vector in $\mathbb{R}^4$ [72]. This view of quaternions is also suggested by how we initially defined the quaternion as

$$\mathbf{q} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

in Eq. (4.33). **Thus, in most practical applications, we simply treat quaternions as column vectors (with some caveats).**

For example, we take the transpose of a quaternion as

$$\mathbf{q}^T = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}$$

so the inner product of two quaternions $\mathbf{p}, \mathbf{q} \in \mathbb{H}$ is

$$\mathbf{p}^T \mathbf{q} = \begin{bmatrix} p_0 & p_1 & p_2 & p_3 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = p_0 q_0 + p_1 q_1 + p_2 q_2 + p_3 q_3$$

4.5.2   Quaternion Properties and Definitions

Conjugate

The **conjugate** of a quaternion is defined as [24, 72]

$$\mathbf{q}^* = \begin{bmatrix} q_0 \\ -\mathbf{q}_v \end{bmatrix} = \begin{bmatrix} q_0 \\ -q_1 \\ -q_2 \\ -q_3 \end{bmatrix} \tag{4.35}$$

Algorithm 32: `quatconj`
Conjugate of a quaternion.

Inputs:
- $\mathbf{q} \in \mathbb{H}$ - quaternion

Procedure:

$$\mathbf{q}^* = \begin{bmatrix} q_0 \\ -q_1 \\ -q_2 \\ -q_3 \end{bmatrix}$$

return $\mathbf{q}^*$

Outputs:
- $\mathbf{q}^{-1} \in \mathbb{H}$ - quaternion conjugate

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- $\mathbf{q}$ does not have to be input as a unit quaternion (this algorithm is for all quaternions, not just unit quaternions).
- $\mathbf{q}^*$ is not normalized.

Test Cases:
- See Appendix A.1.3.

Multiplication

The **Hamilton product** (multiplication) of two quaternions, $\mathbf{p} = (p_0, p_1, p_2, p_3)^T$ and $\mathbf{q} = (q_0, q_1, q_2, q_3)^T$, is defined as

$$\mathbf{p} \otimes \mathbf{q} = \begin{bmatrix} p_0 q_0 - p_1 q_1 - p_2 q_2 - p_3 q_3 \\ p_0 q_1 + p_1 q_0 + p_2 q_3 - p_3 q_2 \\ p_0 q_2 - p_1 q_3 + p_2 q_0 + p_3 q_1 \\ p_0 q_3 + p_1 q_2 - p_2 q_1 + p_3 q_0 \end{bmatrix} \tag{4.36}$$

Note that like matrix multiplication, quaternion multiplication is *not* commutative [72, 80].

$$\mathbf{p} \otimes \mathbf{q} \neq \mathbf{q} \otimes \mathbf{p}$$

However, quaternions are associative over multiplication [91, p. 47]:

$$\boxed{(\mathbf{p} \otimes \mathbf{q}) \otimes \mathbf{r} = \mathbf{p} \otimes (\mathbf{q} \otimes \mathbf{r})} \tag{4.37}$$

Algorithm 33: `quatmul`
Quaternion multiplication (Hamilton product).

Inputs:
- $\mathbf{p} \in \mathbb{H}$ - quaternion
- $\mathbf{q} \in \mathbb{H}$ - quaternion

Procedure:

1. Evaluate the Hamilton product.

$$\mathbf{r} = \begin{bmatrix} p_0 q_0 - p_1 q_1 - p_2 q_2 - p_3 q_3 \\ p_0 q_1 + p_1 q_0 + p_2 q_3 - p_3 q_2 \\ p_0 q_2 - p_1 q_3 + p_2 q_0 + p_3 q_1 \\ p_0 q_3 + p_1 q_2 - p_2 q_1 + p_3 q_0 \end{bmatrix}$$

2. Ensure that the scalar part of the quaternion is positive.

> **if** $r_0 < 0$
> $\quad|\quad$ $\mathbf{r} = -\mathbf{r}$
> **end**

3. Return the result.

> **return r**

Outputs:
   - $\mathbf{r} \in \mathbb{H}$ - Hamilton product of $\mathbf{p}$ and $\mathbf{q}$

Note:
   - This algorithm assumes the scalar-first convention for quaternions.
   - $\mathbf{p}$ and $\mathbf{q}$ do not have to be input as unit quaternions (this algorithm is for all quaternions, not just unit quaternions).
   - $\mathbf{r}$ is not normalized.
   - The scalar part of $\mathbf{r}$ is chosen to be positive.

Test Cases:
   - See Appendix A.1.3.

Norm

The **norm** of a quaternion (representing its magnitude) is defined as

$$\|\mathbf{q}\| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$$

Treating the quaternion as a column vector, we can write its norm as

$$\boxed{\|\mathbf{q}\| = \sqrt{\mathbf{q}^T \mathbf{q}} = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}} \tag{4.38}$$

This is analogous to the 2-norm of a vector in $\mathbb{R}^4$ [72], [46, p. 22].

A **unit quaternion** is a quaternion with a norm of 1.

Algorithm 34: `quatnorm`
Norm of a quaternion.

Inputs:
   - $\mathbf{q} \in \mathbb{H}$ - quaternion

Procedure:

$$\|\mathbf{q}\| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$$

   return $\|\mathbf{q}\|$

Outputs:

  - $\|\mathbf{q}\| \in \mathbb{H}$ - norm of the quaternion

Test Cases:

  - See Appendix A.1.3.

In many programming languages, it is a bit redundant to implement Algorithm 34; for example, we could just use MATLAB's built-in `norm` function, which calculates the 2-norm of a vector by default. However, it is still useful to implement this algorithm so that "under-the-hood" we can replace how we compute $\|\mathbf{q}\|$ is needed.

Inverse

The **inverse** of a quaternion is just its conjugate divided by its norm squared [24], [72], [46, p. 22].

$$\mathbf{q}^{-1} = \frac{\mathbf{q}^*}{\|\mathbf{q}\|^2} = \frac{\mathbf{q}^*}{\mathbf{q}^T\mathbf{q}} = \frac{\mathbf{q}^*}{q_0^2 + q_1^2 + q_2^2 + q_3^2} \qquad (4.39)$$

Note that the inverse of a unit quaternion is just its conjugate (since its norm is 1).

Algorithm 35: `quatinv`
Inverse of a quaternion.

Inputs:

  - $\mathbf{q} \in \mathbb{H}$ - quaternion

Procedure:

   $\mathbf{q}^{-1} = \dfrac{\texttt{quatconj}(\mathbf{q})}{\mathbf{q}^T\mathbf{q}}$ (Algorithm 32)

   return $\mathbf{q}^{-1}$

Outputs:

  - $\mathbf{q}^* \in \mathbb{H}$ - quaternion inverse

Note:

  - This algorithm assumes the scalar-first convention for quaternions.
  - $\mathbf{q}$ does not have to be input as a unit quaternion (this algorithm is for all quaternions, not just unit quaternions).
  - $\mathbf{q}^{-1}$ is not normalized.

Test Cases:

  - See Appendix A.1.3.

Inverse of a Product

The inverse of the Hamilton product of two quaternions is given by the Hamilton product of their inverses, in reverse order (similar to the inverse of the product of two square matrices) [91, p. 48].

$$(\mathbf{p} \otimes \mathbf{q})^{-1} = \mathbf{q}^{-1} \otimes \mathbf{p}^{-1} \tag{4.40}$$

Identity Quaternion

A quaternion multiplied by its inverse is equal to the **identity quaternion**. The identity quaternion is the unit quaternion with scalar part equal to 1 and vector part equal to $\mathbf{0}$. Note that the Hamilton product of a quaternion with its inverse[5] yields the identity quaternion [46, p. 22].

$$\mathbf{q}^{-1} \otimes \mathbf{q} = \mathbf{q} \otimes \mathbf{q}^{-1} = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.41}$$

which results in no rotation.

Normalization

In aerospace simulations, quaternions will often form part of the state vector that is being numerically integrated. Since a standard numerical integrator won't automatically preserve the unit norm constraint on unit quaternion, we will want to regularly renormalize the quaternion. This can be done by simply dividing the quaternion by its norm.

$$\mathbf{q} = \frac{\mathbf{q}}{\|\mathbf{q}\|} \tag{4.42}$$

Since we will need to normalize the quaternion often, we introduce Algorithm 36 in order to reduce boilerplate code.

> In general, we will assume the quaternions input to any algorithms are already normalized; it is *not* the responsibility of an algorithm to normalize its inputs. However, it *is* the responsibility of an algorithm to normalize its *outputs*.

Algorithm 36: `quatnormalize`
Normalize a quaternion.

Inputs:

- $\mathbf{q} \in \mathbb{H}$ - quaternion

Procedure:

$\mathbf{q} = \dfrac{\mathbf{q}}{\texttt{quatnorm}(\mathbf{q})}$ using Algorithm 34

return $\mathbf{q}$

---

[5]   Hence, the quaternion inverse we define is a *multiplicative* inverse

Outputs:
- $\mathbf{q} \in \mathbb{H}$ - unit quaternion

Test Cases:
- See Appendix A.1.3.

4.5.3 Rotations via Unit Quaternions

The rotation from coordinate frame A to coordinate frame B can be parameterized as the unit quaternion

$$\mathbf{q}_{A \to B}$$

Since a unit quaternion multiplied by its inverse results in no rotation, and since the inverse of a unit quaternion is just its conjugate, we know [91, p. 50]

$$\boxed{\mathbf{q}_{B \to A} = \mathbf{q}_{A \to B}^{-1} = \mathbf{q}_{A \to B}^{*}} \tag{4.43}$$

Applying a Single Rotation

Let $[r]_A$ be a vector expressed in frame A. Let

$$\boxed{\mathbf{p}_A = \begin{pmatrix} 0 \\ [r]_A \end{pmatrix}} \tag{4.44}$$

To express this vector in frame B[6] [91, p. 50],

$$\boxed{\mathbf{p}_B = \mathbf{q}_{A \to B}^{-1} \otimes \mathbf{p}_A \otimes \mathbf{q}_{A \to B}} \tag{4.45}$$

where

$$\boxed{\mathbf{p}_B = \begin{pmatrix} 0 \\ [r]_B \end{pmatrix}} \tag{4.46}$$

If we let

$$\mathbf{q}_{A \to B} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

then using the definition of the Hamilton product from Eq. (4.36), we can simply Eq. (4.45) to [73]

$$[r]_B = \underbrace{\begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2 q_3 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_3 - q_0 q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix}}_{\mathbf{R}_{A \to B}} [r]_A$$

---

[6] [73] provides this equation as

$$[r]_B = \mathbf{q}_{A \to B} \otimes \mathbf{p}_A \otimes \mathbf{q}_{A \to B}^{-1}$$

, but from the discussion in that article, this equation appears to encode an active rotation, rather than the passive rotations we use in aerospace. Even more confusingly, [46, p. 34], which uses passive rotations throughout almost all rotation discussions, also uses this active version of the quaternion rotation. However, [80] does clearly indicate the active vs. passive formulations, and [91, p. 50] also defines *and* describes the passive formulation.

In practice, we first convert the quaternion to a rotation matrix (see Algorithm 39 in Section 4.5.4), and then perform the rotation using that rotation matrix. This results in fewer total arithemtic operations than using Algorithm 33 twice to evaluate Eq. (4.45) directly [24], [73].

Algorithm 37: `quatrotate`
Passive rotation of a vector by a unit quaternion.

Inputs:
- $[\mathbf{q}_{A \to B} \in \mathbb{H}$ - unit quaternion representing rotation from frame A to frame B
- $[\boldsymbol{r}]_A \in \mathbb{R}^3$ - vector expressed in coordinate frame A

Procedure:

$[\boldsymbol{r}]_B = \texttt{matrotate}\,(\texttt{quat2mat}(\mathbf{q}_{A \to B}), [\boldsymbol{r}]_A)$ (Algorithms 8 and 39)

return $[\boldsymbol{r}]_B$

Outputs:
- $[\boldsymbol{r}]_B \in \mathbb{R}^3$ - vector expressed in coordinate frame B

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- This algorithm normalizes the input quaternion.

Test Cases:
- See Appendix A.1.3.

Chaining Rotations

Let's define the following three unit quaternions:

1. $q_{A \to B}$: unit quaternion representing the rotation from frame A to frame B
2. $q_{B \to C}$: unit quaternion representing the rotation from frame B to frame C
3. $q_{A \to C}$: unit quaternion representing the rotation from frame A to frame C

Recall that for rotation matrices, the rotations chain together right-to-left in the order of application:

$$\mathbf{R}_{A \to C} = \mathbf{R}_{B \to C}\mathbf{R}_{A \to B}$$

For unit quaternions, the rotations chain together in the opposite order, from left-to-right [24, 73]:

$$\boxed{\mathbf{q}_{A \to C} = (\mathbf{q}_{A \to B}) \otimes (\mathbf{q}_{B \to C})} \tag{4.47}$$

This can be shown by first applying the rotation from A to B using Eq. (4.45).

$$\mathbf{p}_B = \mathbf{q}_{A \to B}^{-1} \otimes \mathbf{p}_A \otimes \mathbf{q}_{A \to B}$$

Next, using Eq. (4.45) once more to apply the rotation from B to C,

$$
\begin{aligned}
\mathbf{p}_C &= \mathbf{q}_{B \to C}^{-1} \otimes \mathbf{p}_B \otimes \mathbf{q}_{B \to C} \\
&= \mathbf{q}_{B \to C}^{-1} \otimes (\mathbf{q}_{A \to B}^{-1} \otimes \mathbf{p}_A \otimes \mathbf{q}_{A \to B}) \otimes \mathbf{q}_{B \to C} \\
&= (\mathbf{q}_{B \to C}^{-1} \otimes \mathbf{q}_{A \to B}^{-1}) \otimes \mathbf{p}_A \otimes (\mathbf{q}_{A \to B} \otimes \mathbf{q}_{B \to C}) \\
&= (\mathbf{q}_{A \to B} \otimes \mathbf{q}_{B \to C})^{-1} \otimes \mathbf{p}_A \otimes (\mathbf{q}_{A \to B} \otimes \mathbf{q}_{B \to C})
\end{aligned}
\tag{4.48}
$$

Note that the simplification above relies on Eqs. (4.40) and (4.37). From Eq. (4.45), we also know

$$\mathbf{p}_C = \mathbf{q}_{A\to C}^{-1} \otimes \mathbf{p}_A \otimes \mathbf{q}_{A\to C} \tag{4.49}$$

Comparing Eqs. (4.48) and (4.49), we can see that

$$\mathbf{q}_{A\to C} = (\mathbf{q}_{A\to B}) \otimes (\mathbf{q}_{B\to C})$$

as given previously by Eq. (4.47) [91, p. 50], [80].

Due to floating point errors, the Hamilton product of two unit quaternions could have a magnitude that is slightly different than 1. Since in this context we are using quaternions exclusively for parameterizing rotations, we normalize the Hamilton product of two unit quaternions to ensure that the result is a unit quaternion.

Algorithm 38: `quatchain`
Chaining passive rotations represented by unit quaternions.

Inputs:
- $\mathbf{q}_{A\to B} \in \mathbb{H}$ - unit quaternion representing rotation from frame A to frame B
- $\mathbf{q}_{B\to C} \in \mathbb{H}$ - unit quaternion representing rotation from frame B to frame C

Procedure:
1. Evaluate the Hamilton product to find the chained rotation (Algorithm 33).

    $$\mathbf{q}_{A\to C} = \texttt{quatmul}(\mathbf{q}_{A\to B}, \mathbf{q}_{B\to C})$$

2. Normalize the result (Algorithm 36).

    $$\mathbf{q}_{A\to C} = \texttt{quatnormalize}(\mathbf{q}_{A\to C})$$

3. Return the result.

    return $\mathbf{q}_{A\to C}$

Outputs:
- $\mathbf{q}_{A\to C} \in \mathbb{H}$ - unit quaternion representing rotation from frame A to frame C

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- This algorithm assumes that $\mathbf{q}_{A\to B}$ and $\mathbf{q}_{B\to C}$ are input as unit quaternions, so it does *not* normalize them.
- $\mathbf{q}_{A\to C}$ is normalized.
- The scalar part of $\mathbf{q}_{A\to C}$ is chosen to be positive.

Test Cases:
- See Appendix A.1.3.

4.5.4 Relationship With Rotation Matrices

Recall that we can take a vector expressed in frame A and resolve it in frame B using a rotation matrix.

$$[\mathbf{v}]_B = \mathbf{R}_{A\to B}[\mathbf{v}]_A$$

Let $\mathbf{q}$ be a quaternion representing a rotation. The corresponding rotation matrix, $\mathbf{R}$, can be determined as [73], [46, p. 23]

$$\mathbf{R} = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2 q_3 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_3 - q_0 q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix} \tag{4.50}$$

If we are storing a rotation as a quaternion, we first convert the quaternion to a rotation matrix using Algorithm 39 below. We can then apply the rotation using the rotation matrix.

Algorithm 39: `quat2mat`
Unit quaternion to rotation matrix (passive rotation).

Inputs:
- $\mathbf{q} \in \mathbb{H}$ - unit quaternion

Procedure:
1. Normalize the quaternion (Algorithm 36).

$$\mathbf{q} = \texttt{quatnormalize}(\mathbf{q})$$

2. Store the elements of the unit quaternion, $\mathbf{q} = (q_0, q_1, q_2, q_3)^T$, individually.
3. Construct the rotation matrix.

$$\mathbf{R} = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2 q_3 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_3 - q_0 q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix}$$

4. Return the result.

return $\mathbf{R}$

Outputs:
- $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ - rotation matrix

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- This algorithm normalizes the input quaternion.

Test Cases:
- See Appendix A.1.2.

Extracting a quaternion from a rotation matrix is a more involved process since there are many alternative ways that each quaternion element can be computed. While all alternatives are algebraically equivalent, they are not numerically equivalent. As an example, one way we could compute that elements would be to first compute q_1 as

$$q_1 = \pm \frac{1}{2} \sqrt{1 + R_{11} - R_{22} - R_{33}}$$

and then compute the remaining elements as

$$q_0 = \frac{R_{23} - R_{32}}{4q_1}$$

$$q_2 = \frac{R_{12} + R_{21}}{4q_1}$$

$$q_3 = \frac{R_{31} + R_{13}}{4q_1}$$

However, if q_1 is near 0, then the problem is ill-conditioned mathematically. The most popular and common method is **Shepperd's method**, which first computes each quaternion element with an equation similar to the one for q_1, picks the element that has the largest value, and then computes the remaining elements using equations similar to the ones for q_0, q_2, and q_3 above. Shepperd's method is covered in more detail in [95, pp. 24–26], [81], [82], and [9].

While Shepperd's method considers 4 alternatives, [82] introduced the Sarabandi-Thomas method in 2019 that considers 16 alternatives, and is (a) more efficient computationally and (b) is more accurate. In that same year, Sarabandi and Thomas also performed a survey on the computation of quaternions from rotation matrices in [81] and determined that the best method is **Cayley's method**. This method is the fastest, has the lowest average error, and is the simplest to implement (the other methods typically require either computing multiple solutions and using a voting scheme, or using if/else statements when computing each quaternion element. Note that the worst case error with Cayley method was slightly higher than that of the Sarabandi-Thomas and Shepperd methods.

Cayley's method computes the unit quaternion as

$$\mathbf{q} = \frac{1}{4} \begin{bmatrix} \sqrt{(R_{11} + R_{22} + R_{33} + 1)^2 + (R_{32} - R_{23})^2 + (R_{13} - R_{31})^2 + (R_{21} - R_{12})^2} \\ \text{sgn}\,(R_{23} - R_{32})\sqrt{(R_{32} - R_{23})^2 + (R_{11} - R_{22} - R_{33} + 1)^2 + (R_{21} + R_{12})^2 + (R_{31} + R_{13})^2} \\ \text{sgn}\,(R_{31} - R_{13})\sqrt{(R_{13} - R_{31})^2 + (R_{21} + R_{12})^2 + (R_{22} - R_{11} - R_{33} + 1)^2 + (R_{32} + R_{23})^2} \\ \text{sgn}\,(R_{12} - R_{21})\sqrt{(R_{21} - R_{12})^2 + (R_{31} + R_{13})^2 + (R_{32} + R_{23})^2 + (R_{33} - R_{11} - R_{22} + 1)^2} \end{bmatrix} \tag{4.51}$$

Note that the scalar part already follows the quaternion sign convention defined by Convention 37 (i.e. $q_0 \geq 0$). Additionally, note that we the signs on the vector part are reversed from what [81] defines[7]. We implement Cayley's method for computing the quaternion from a rotation matrix in Algorithm 40 below.

Algorithm 40: `mat2quat`
Rotation matrix to unit quaternion (passive rotation).

Inputs:
- $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ - rotation matrix

Procedure:
1. Compute the quaternion using Eq. (4.51). Note that we don't need to include the coefficient of $1/4$ since we will be normalizing in the next step anyways.
2. Normalize the quaternion (Algorithm 36).

$$\mathbf{q} = \texttt{quatnormalize}(\mathbf{q})$$

3. Return the result.

return q

Outputs:
- $\mathbf{q} \in \mathbb{H}$ - unit quaternion

---

[7] This flip in sign was determined through testing Algorithm 40.

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- The scalar part, q_0, is chosen to be positive.
- **q** is normalized.

Test Cases:
- See Appendix A.1.2.

4.5.5 Relationship With Euler Angles

Recall Eqs. (4.17), (4.18), and (4.19) defining the 3-2-1 Euler angles in terms of the elements of the rotation matrix. Also, recall Eq. (4.50) defining the elements of the rotation matrix in terms of a unit quaternion. Using these equations, we can obtain the 3-2-1 Euler angles corresponding to a unit quaternion [46, p. 26]

$$\psi = \begin{cases} \arctan2\left[2(q_1q_2 + q_0q_3), 1 - 2(q_2^2 + q_3^2)\right], & |R_{13}| < 1 \\ 0, & |R_{13}| \geq 1 \end{cases} \tag{4.52}$$

$$\theta = \begin{cases} -\arcsin R_{13}, & |R_{13}| < 1 \\ -\left(\dfrac{\pi}{2}\right)\operatorname{sgn} R_{13}, & |R_{13}| \geq 1 \end{cases} \tag{4.53}$$

$$\phi = \begin{cases} \arctan2\left[2(q_2q_3 + q_0q_1), 1 - 2(q_1^2 + q_2^2)\right], & |R_{13}| < 1 \\ \arctan2\left[2(q_1q_2 - q_0q_3), 2(q_1q_3 + q_0q_2)\right], & R_{13} \leq -1 \\ -\arctan2\left[2(q_2q_3 - q_0q_1), 1 - 2(q_1^2 + q_3^2)\right], & R_{13} \geq 1 \end{cases} \tag{4.54}$$

where

$$R_{13} = 2(q_1q_3 - q_0q_2) \tag{4.55}$$

Algorithm 41 below is identical in logic to that of Algorithm 27. In principle, we are converting the quaternion to a rotation matrix, and then extracting the 3-2-1 Euler angles from the rotation matrix.

Algorithm 41: `quat2eul_321`
Unit quaternion to 3-2-1 Euler angles (yaw, pitch, and roll) (passive rotation).

Inputs:
- $\mathbf{q} \in \mathbb{H}$ - unit quaternion

Procedure:
1. Normalize the quaternion (Algorithm 36).

$$\mathbf{q} = \texttt{quatnormalize}(\mathbf{q})$$

2. Store the elements of the unit quaternion, $\mathbf{q} = (q_0, q_1, q_2, q_3)^T$, individually.
3. Calculate the R_{13} element of the rotation matrix.

$$R_{13} = 2(q_1q_3 - q_0q_2)$$

4. Booleans that store whether or not we are in a singular case, and if so, which one.

$$\texttt{pitch\_up} = (R_{13} \leq -1)$$
$$\texttt{pitch\_down} = (R_{13} \geq 1)$$
$$\texttt{singular} = (\texttt{pitch\_up or pitch\_down})$$

5. Yaw angle [rad].

   if `singular`
   | $\psi = 0$
   else
   | $\psi = \arctan2\left[2(q_1 q_2 + q_0 q_3), 1 - 2(q_2^2 + q_3^2)\right]$
   end

6. Pitch angle [rad].

   if `singular`
   | $\theta = -\left(\dfrac{\pi}{2}\right) \operatorname{sgn} R_{13}$
   else
   | $\theta = -\arcsin R_{13}$
   end

7. Roll angle [rad].

   if `pitch_up`
   | $\phi = \arctan2\left[2(q_1 q_2 - q_0 q_3), 2(q_1 q_3 + q_0 q_2)\right]$
   else if `pitch_down`
   | $\phi = -\arctan2\left[2(q_2 q_3 - q_0 q_1), 1 - 2(q_1^2 + q_3^2)\right]$
   else
   | $\phi = \arctan2\left[2(q_2 q_3 + q_0 q_1), 1 - 2(q_1^2 + q_2^2)\right]$
   end

8. Return the results.

   return ψ, θ, ϕ

Outputs:
- $\psi \in \mathbb{R}$ - yaw angle (1st rotation, about 3rd axis) [rad]
- $\theta \in \mathbb{R}$ - pitch angle (2nd rotation, about 2nd axis) [rad]
- $\phi \in \mathbb{R}$ - roll angle (3rd rotation, about 1st axis) [rad]

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- This algorithm normalizes the input quaternion.
- $\psi \in (-\pi, \pi]$
- $\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$
- $\phi \in (-\pi, \pi]$
- When a singularity corresponding to the pitch angles of $\theta = \pm\pi/2$ is encountered, the yaw angle (ψ) is set to 0 and the roll angle (ϕ) is determined accordingly.

Test Cases:
- See Appendix A.1.2.

A unit quaternion can be constructed from the 3-2-1 Euler angles as [91, p. 52], [28, p. 12]

$$
\mathbf{q} = \begin{bmatrix} \cos\left(\dfrac{\phi}{2}\right)\cos\left(\dfrac{\theta}{2}\right)\cos\left(\dfrac{\psi}{2}\right) + \sin\left(\dfrac{\phi}{2}\right)\sin\left(\dfrac{\theta}{2}\right)\sin\left(\dfrac{\psi}{2}\right) \\[2mm] \sin\left(\dfrac{\phi}{2}\right)\cos\left(\dfrac{\theta}{2}\right)\cos\left(\dfrac{\psi}{2}\right) - \cos\left(\dfrac{\phi}{2}\right)\sin\left(\dfrac{\theta}{2}\right)\sin\left(\dfrac{\psi}{2}\right) \\[2mm] \cos\left(\dfrac{\phi}{2}\right)\sin\left(\dfrac{\theta}{2}\right)\cos\left(\dfrac{\psi}{2}\right) + \sin\left(\dfrac{\phi}{2}\right)\cos\left(\dfrac{\theta}{2}\right)\sin\left(\dfrac{\psi}{2}\right) \\[2mm] \cos\left(\dfrac{\phi}{2}\right)\cos\left(\dfrac{\theta}{2}\right)\sin\left(\dfrac{\psi}{2}\right) - \sin\left(\dfrac{\phi}{2}\right)\sin\left(\dfrac{\theta}{2}\right)\cos\left(\dfrac{\psi}{2}\right) \end{bmatrix}
\tag{4.56}
$$

Algorithm 42: `eul2quat_321`
3-2-1 Euler angles (yaw, pitch, and roll) to unit quaternion (passive rotation).

Inputs:
- $\psi \in \mathbb{R}$ - yaw angle (1st rotation, about 3rd axis) $[\mathrm{rad}]$
- $\theta \in \mathbb{R}$ - pitch angle (2nd rotation, about 2nd axis) $[\mathrm{rad}]$
- $\phi \in \mathbb{R}$ - roll angle (3rd rotation, about 1st axis) $[\mathrm{rad}]$

Procedure:
1. Precompute the trigonometric functions.

$$
c_1 = \cos\left(\frac{\phi}{2}\right)
$$

$$
s_1 = \sin\left(\frac{\phi}{2}\right)
$$

$$
c_2 = \cos\left(\frac{\theta}{2}\right)
$$

$$
s_2 = \sin\left(\frac{\theta}{2}\right)
$$

$$
c_3 = \cos\left(\frac{\psi}{2}\right)
$$

$$
s_3 = \sin\left(\frac{\psi}{2}\right)
$$

2. Compute the quaternion.

$$
\mathbf{q} = \begin{bmatrix} c_1 c_2 c_3 + s_1 s_2 s_3 \\ s_1 c_2 c_3 - c_1 s_2 s_3 \\ c_1 s_2 c_3 + s_1 c_2 s_3 \\ c_1 c_2 s_3 - s_1 s_2 c_3 \end{bmatrix}
$$

3. Normalize the quaternion (Algorithm 36).

$$
\mathbf{q} = \texttt{quatnormalize}(\mathbf{q})
$$

4. Return the result.

> **return q**

Outputs:
- $\mathbf{q} \in \mathbb{H}$ - unit quaternion

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- The scalar part, q_0, is chosen to be positive.
- $\mathbf{q}$ is normalized.

Test Cases:
- See Appendix A.1.2.

4.5.6 Relationship With Axis-Angle Representation

Given a principal rotation vector, $\mathbf{e} = (e_1, e_2, e_3)^T$, and principal angle, Φ, we can define the unit quaternion as [95, p. 23], [80]

$$\mathbf{q} = \begin{bmatrix} \cos\left(\dfrac{\Phi}{2}\right) \\ \sin\left(\dfrac{\Phi}{2}\right)\mathbf{e} \end{bmatrix} \begin{bmatrix} \cos\left(\dfrac{\Phi}{2}\right) \\ e_1 \sin\left(\dfrac{\Phi}{2}\right) \\ e_2 \sin\left(\dfrac{\Phi}{2}\right) \\ e_3 \sin\left(\dfrac{\Phi}{2}\right) \end{bmatrix} \tag{4.57}$$

Recall Convention 37, which defined the sign convention for quaternions such that q_0 is always positive. In this case, if $q_0 < 0$, we must switch the sign on the entire quaternion.

Algorithm 43: `axang2quat`
Axis-angle representation to unit quaternion (passive rotation).

Inputs:
- $\mathbf{e} \in \mathbb{R}^3$ - principal rotation vector
- $\Phi \in \mathbb{R}$ - principal angle [rad]

Procedure:
1. Normalize the principal rotation vector.

$$\mathbf{e} = \frac{\mathbf{e}}{\|\mathbf{e}\|}$$

2. Determine the quaternion.

$$\mathbf{q} = \begin{bmatrix} \cos\left(\dfrac{\Phi}{2}\right) \\ \sin\left(\dfrac{\Phi}{2}\right)\mathbf{e} \end{bmatrix}$$

3. Ensure that the scalar part of the quaternion is positive.

if $q_0 < 0$
$\quad \mid \quad$ **q** = −**q**
end

4. Normalize the quaternion (Algorithm 36).

$\mathbf{q} = \texttt{quatnormalize}(\mathbf{q})$

5. Return the result.

return q

Outputs:

- $\mathbf{q} \in \mathbb{H}$ - unit quaternion

Note:

- This algorithm normalizes **e**.
- This algorithm assumes the scalar-first convention for quaternions.
- The scalar part, q_0, is chosen to be positive.
- **q** is normalized.

Test Cases:

- See Appendix A.1.2.

To obtain the Euler axis and angle from a unit quaternion, we can just simply solve Eq. (4.57) for **e** and Φ. First, solving for Φ,

$$\boxed{\Phi = 2\arccos q_0} \tag{4.58}$$

Note that Eq. (4.58) will always return a positive Φ when using the quaternion sign convention defined by Convention 37. However, in practice, we need to protect against the case where this sign convention is not followed, and take the precaution of checking the sign on q_0 and negating the quaternion if needed. Additionally, due to finite precision arithmetic, q_0 may sometimes be slightly less than −1 or slightly greater than 1, which will cause issues with the inverse cosine function. Thus, we must manually clamp q_0 to have a value between −1 and 1; this is the approach that [17] takes.

However, this approach can lead to imprecise results near $\pm - 1$ due to the inverse cosine losing numerical precision near the endpoints of its domain. This occurs due to large derivatives near these endpoints, causing it to be very sensitive to small numerical disturbances [10, 66]. An alternative approach is to use

$$\boxed{\Phi = 2\arctan2\left(\|\mathbf{q}_v\|, q_0\right)} \tag{4.59}$$

where $\mathbf{q}_v = (q_1, q_2, q_3)^T$ is the vector part of the quaternion [10, 24]. Since $\|\mathbf{q}_v\| \geq 0$ by definition, and since $q_0 \geq 0$ when following the quaternion sign convention given by Convention 37, we know that

$$0 \leq \arctan2\left(\|\mathbf{q}_v\|, q_0\right) \leq \frac{\pi}{2}$$

which implies that

$$0 \leq \Phi \leq \pi$$

as desired. Not only is this approach more numerically stable while also returning the angle in the correct domain, but it also avoids the extra logic of clamping q_0 to ensure the input to the inverse cosine function is within its domain.

Next, we can solve for **e** as

$$\mathbf{e} = \frac{1}{\sin\left(\Phi/2\right)} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{4.60}$$

However, note that

$$\sin^2\left(\frac{\Phi}{2}\right) + \cos^2\left(\frac{\Phi}{2}\right) = 1 \quad \rightarrow \quad \sin\left(\frac{\Phi}{2}\right) = \sqrt{1 - \cos^2\left(\frac{\Phi}{2}\right)} = \sqrt{1 - q_0^2}$$

Substituting this into Eq. (4.60),

$$\mathbf{e} = \frac{1}{\sqrt{1 - q_0^2}} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{4.61}$$

At this point, we still have an issue; recall that the axis-angle representation has a singularity at $\Phi = 0$. We can see this immediately from Eq. (4.60), and from Eq. (4.61), we can see that this also corresponds to the case where $|q_0| = 1$. Since this singularity corresponds to the case where there is no rotation, we can set the principal rotation vector to $\mathbf{e} = (1, 0, 0)^T$ by default, as we did for this same singularity in Section 4.4.1 previously.

$$\mathbf{e} = \begin{cases} \dfrac{1}{\sqrt{1 - q_0^2}} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}, & |q_0| < 1 \\[12pt] (1, 0, 0)^T, & q_0 = 1 \end{cases} \tag{4.62}$$

Note that in its implementation, we do not need to compute the coefficient on **e** in Eq. (4.62), since we will normalize the principal rotation vector anyways.

Algorithm 44: `quat2axang`
Quaternion to axis-angle representation (passive rotation).

Inputs:
- $\mathbf{q} \in \mathbb{H}$ - quaternion

Procedure:
1. Ensure that the scalar part of the quaternion is positive.

   if $q_0 < 0$
       $\mathbf{q} = -\mathbf{q}$
   end

2. Normalize the quaternion (Algorithm 36).

   $\mathbf{q} = \mathtt{quatnormalize}(\mathbf{q})$

3. Principal rotation vector.

   if $|q_0| < 1$

       $\mathbf{e} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}$

else

$$\mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

end

4. Normalize the principal rotation vector.

$$\mathbf{e} = \frac{\mathbf{e}}{\|\mathbf{e}\|}$$

5. Vector part of the quaternion.

$$\mathbf{q}_v = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

6. Principal angle [rad].

$$\Phi = 2 \arctan2 \left(\|\mathbf{q}_v\|, q_0 \right)$$

7. Return the result.

return $\mathbf{e}, \Phi$

Outputs:
- $\mathbf{e} \in \mathbb{R}^3$　　- principal rotation vector
- $\Phi \in \mathbb{R}$　　- principal angle [rad]

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- This algorithm normalizes the input quaternion.
- $\|\mathbf{e}\| = 1$.
- $\Phi \in [0, \pi]$
- If $\Phi = 0$, then $\mathbf{e}$ is returned as $(1, 0, 0)^T$.

Test Cases:
- See Appendix A.1.2.

4.5.7　Angle Between Unit Quaternions

In general, a quaternion can be viewed as a four-dimensional vector (as discussed in Section 4.5.1), representing a point in a four-dimensional space. However, constraining it to a unit magnitude yields a three dimensional space equivalent to the surface a hypersphere. Alternatively, you can view it as taking the unit vector $(1, 0, 0)^T$, and performing every possible rotation of that unit vector; its tip will trace out the surface of the unit sphere. Thus, a single quaternion can be visualized as a point on the unit sphere resulting from applying a rotation to a unit vector.

　　　　The angle between two quaternions is then the angle that subtends the great arc connecting the two corresponding points on the unit sphere [73, 84, 88]. Consider two quaternions $\mathbf{q}_1, \mathbf{q}_2 \in \mathbb{H}$. The angle between them can be

computed as [24][8,9]

$$\theta = 4 \arctan2\left(\|\mathbf{q}_1 - \mathbf{q}_2\|, \|\mathbf{q}_1 + \mathbf{q}_2\|\right) \tag{4.63}$$

Since $\|\mathbf{q}_1 - \mathbf{q}_2\| \geq 0$ and $\|\mathbf{q}_1 + \mathbf{q}_2\| \geq 0$, we know that

$$0 \leq \arctan2\left(\|\mathbf{q}_1 - \mathbf{q}_2\|, \|\mathbf{q}_1 + \mathbf{q}_2\|\right) \leq \frac{\pi}{2}$$

Since θ is this value times four, we know

$$0 \leq \theta \leq 2\pi$$

This presents the problem; there are two great arcs connecting two orientations on a unit sphere; one subtended by an angle less than π, and another subtended by an angle greater than π. These two solutions, θ_1 and θ_2, are simply related by

$$\theta_2 = 2\pi - \theta_1$$

Therefore, if we happen to get the larger of the two angles, we can simply subtract it from 2π to get the shorter of the two angles, and ensure that $0 \leq \theta \leq \pi$. We incorporate this adjustment into Algorithm 45 below for computing the angle between two quaternions.

Algorithm 45: `quatang`
Angle between two unit quaternions.

Inputs:
- $\mathbf{q}_1 \in \mathbb{H}$ - unit quaternion
- $\mathbf{q}_2 \in \mathbb{H}$ - unit quaternion

Procedure:
1. Angle between the unit quaternions (Algorithm 34) [rad].

$$\theta = 4 \arctan2\left[\mathtt{quatnorm}(\mathbf{q}_1 - \mathbf{q}_2), \mathtt{quatnorm}(\mathbf{q}_1 + \mathbf{q}_2)\right]$$

2. Ensure that the shorter of the two arcs is used.

    if $\theta > \pi$
    $\quad\mid\quad \theta = 2\pi - \theta$
    end

3. Return the result.

    return θ

Outputs:
- $\theta \in \mathbb{R}$ - angle between the unit quaternions [rad]

Note:
- This algorithm assumes that $\mathbf{q}_1$ and $\mathbf{q}_2$ are input as unit quaternions, so it does *not* normalize them.
- This algorithm returns the smaller of the two angles between the unit quaternions, such that $\theta \in [0, \pi]$.

---

[8]   Note that [24] actually defines the half angle $(\theta/2) = 2 \arctan2\left(\|\mathbf{q}_1 - \mathbf{q}_2\|, \|\mathbf{q}_1 + \mathbf{q}_2\|\right)$.
[9]   An alternative computation is $\theta = 2 \arccos\left(\mathbf{q}_1^T \mathbf{q}_2\right)$. However, this approach loses numerical precision when the argument of the inverse cosine function nears ± 1 (for the same reason as discussed in the previous section for another procedure using the inverse cosine function). Also note that the four-quadrant inverse tangent can already handle the case where $\|\mathbf{q}_1 + \mathbf{q}_2\| = 0$, whereas performing $\arctan(y/x)$ when $x = 0$ could lead to issues [24], [66].

Test Cases:
- See Appendix A.1.3.

The Axis-Angle Approach

This angle can also be found from axis-angle representation of a rotation. Recall from Section 4.5.3 that

$$\mathbf{q}_{A \to C} = \mathbf{q}_{A \to B} \otimes \mathbf{q}_{B \to C}$$

Let's assume we know $\mathbf{q}_{A \to B}$ and $\mathbf{q}_{A \to C}$, and want to find the intermediate rotation $\mathbf{q}_{B \to C}$. Left-multiplying both sides by $\mathbf{q}_{A \to B}^*$, and noting that $\mathbf{q}_{A \to B}^* = \mathbf{q}_{A \to B}^{-1}$ for unit quaternions,

$$\mathbf{q}_{A \to B}^{-1} \otimes \mathbf{q}_{A \to C} = \mathbf{q}_{A \to B}^{-1} \otimes (\mathbf{q}_{A \to B} \otimes \mathbf{q}_{B \to C}) = (\mathbf{q}_{A \to B}^{-1} \otimes \mathbf{q}_{A \to B}) \otimes \mathbf{q}_{B \to C}$$

$$\therefore \mathbf{q}_{B \to C} = \mathbf{q}_{A \to B}^{-1} \otimes \mathbf{q}_{A \to C}$$

In this case, we have the unit quaternions $\mathbf{q}_1$ and $\mathbf{q}_2$, essentially representing two orientations or attitudes. They are linked via an intermediate rotation, $\mathbf{q}$. Therefore, comparing to the example above, we can make the substitutions $\mathbf{q}_1 = \mathbf{q}_{A \to B}$, $\mathbf{q}_2 = \mathbf{A} \to \mathbf{C}$, and $\mathbf{q} = \mathbf{q}_{B \to C}$.

$$\mathbf{q} = \mathbf{q}_1^{-1} \otimes \mathbf{q}_2$$

The angle between $\mathbf{q}_1$ and $\mathbf{q}_2$ is then just the principal angle Φ of the rotation represented by $\mathbf{q}$. We *could* use the following procedure [5]:

1. Calculate $\mathbf{q}_1^*$ using Algorithm 32.

   $$\mathbf{q}_1^* = \texttt{quatconj}(\mathbf{q}_1)$$

2. Calculate $\mathbf{q}$ using Algorithm 33.

   $$\mathbf{q} = \texttt{quatmul}(\mathbf{q}_1^*, \mathbf{q}_q)$$

3. Find θ using Algorithm 44.

   $$[\sim, \theta] = \texttt{quat2axang}(\mathbf{q})$$

However, it is much more efficient to just use Algorithm 45. Nonetheless, we can use this procedure to unit test Algorithm 45.

4.5.8   Spherical Linear Interpolation (SLERP)

Spherical linear interpolation (SLERP) is a method for interpolating between two unit quaternions. Similar to how linear interpolation interpolates between two points using a line with a constant derivative, SLERP interpolates between two unit quaternions by moving at a constant speed along a great circle arc on the unit sphere [84, 88].

Let $t \in [0, 1]$ be the **interpolation parameter** representing the fraction of the great arc connecting the two unit quaternions at which we want to find a third quaternion via SLERP. Letting $\mathbf{q}_1, \mathbf{q}_2 \in \mathbb{H}$ be two unit quaternions, the basic SLERP formula is given by

$$\mathbf{q}(t) = \left(\frac{\sin\left[(1-t)\phi\right]}{\sin \phi} \right) \mathbf{q}_1 + \left(\frac{\sin t\phi}{\sin \phi} \right) \mathbf{q}_2 \tag{4.64}$$

where ϕ satisfies (treating the unit quaternions as column vectors)

$$\phi = \arccos\left(\mathbf{q}_1^T \mathbf{q}_2\right) \tag{4.65}$$

However, this presents a huge issue, that many resources don't make any mention of. We can recall from a footnote in Section 4.5.7 that the angle between two quaternions can also be computed as

$$\theta = 2 \arccos\left(\mathbf{q}_1^T \mathbf{q}_2\right)$$

which implies that ϕ is the half-angle between the two unit quaternions.

$$\boxed{\phi = \frac{\theta}{2}}$$

(4.66)

The issue with this is that there are two great arcs connecting the points on the unit sphere represented by $\mathbf{q}_1$ and $\mathbf{q}_2$. Algorithm 45 already accounts for this and is written to return the smaller of the two angles (i.e. the angle subtending the shorter of the two great arcs). [96] and [24] both include slightly different procedures for picking the shortest path. Here, we use somewhat of a hybrid of the two. First, we can use Algorithm 45 to find the smaller of the two angles between the unit quaternions, and then calculate ϕ as half of that angle (as given by Eq. (4.66)). Then, we perform a check on the sign of $\mathbf{q}_1^T \mathbf{q}_2$; if $\mathbf{q}_1^T \mathbf{q}_2 < 0$, then we will have to interpolate in the opposite direction as given by Eq. (4.64) (i.e. the second term will have to be negative)[10].

Algorithm 46: `quatslerp`
Spherical linear interpolation (SLERP) between two unit quaternions.

Inputs:
- $\mathbf{q}_1 \in \mathbb{H}$ - unit quaternion
- $\mathbf{q}_2 \in \mathbb{H}$ - unit quaternion
- t - interpolation parameter between 0 and 1 (inclusive)

Procedure:
1. Half-angle between the unit quaternions (Algorithm 45) [rad].

$$\phi = \frac{\texttt{quatang}(\mathbf{q}_1, \mathbf{q}_2)}{2}$$

2. Evaluate the trigonometric functions.

$$a = \sin \phi$$
$$b = \sin (t\phi)$$
$$c = \sin [(1 - t)\phi]$$

3. Perform SLERP along the shorter arc.

if $\mathbf{q}_1^T \mathbf{q}_2 \geq 0$

$$\mathbf{q} = \left(\frac{c}{a}\right) \mathbf{q}_1 + \left(\frac{b}{a}\right) \mathbf{q}_2$$

else

$$\mathbf{q} = \left(\frac{c}{a}\right) \mathbf{q}_1 - \left(\frac{b}{a}\right) \mathbf{q}_2$$

end

4. Normalize the result (Algorithm 36).

$$\mathbf{q} = \texttt{quatnormalize}(\mathbf{q})$$

---

[10] Note that [96] actually makes the first term negative. In the end, it doesn't matter, because $\mathbf{q}$ and $-\mathbf{q}$ represent the same rotation.

5. Ensure that the scalar part of the quaternion is positive.

if $q_0 < 0$
$\quad|\quad$ $\mathbf{q} = -\mathbf{q}$
end

6. Return the result.

return q

Outputs:
- $\mathbf{q} \in \mathbb{H}$ - interpolated unit quaternion

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- This algorithm assumes that $\mathbf{q}_1$ and $\mathbf{q}_2$ are input as unit quaternions, so it does *not* normalize them.
- $\mathbf{q}$ is normalized.
- The scalar part of $\mathbf{q}$ is chosen to be positive.

Test Cases:
- See Appendix A.1.3.

4.6   Euler and Body Rates

TODO TODO: euler rates do NOT make up an angular velocity vector

4.7   Quaternion Kinematic Equation

TODO: test cases TODO: code docs Recall the angular velocity of the the body with respect to world frame, expressed in the world frame, from Section 4.6.

$$\left[{}^{\text{body/world}}\boldsymbol{\omega} \right]_{\text{body}} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Let's define the **body rate matrix**, $\boldsymbol{\Omega} \in \mathbb{R}^{4\times4}$, as

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix} \tag{4.67}$$

Let $\mathbf{q}_{\text{world}\rightarrow\text{body}} = (q_0, q_1, q_2, q_3)^T \in \mathbb{H}$ be the unit quaternion representing the passive rotation from the world frame to the body frame. The time derivative of this quaternion is defined as [91, pp. 50–51]

$$\frac{d(\mathbf{q}_{\text{world}\rightarrow\text{body}})}{dt} = \frac{1}{2}\boldsymbol{\Omega}\mathbf{q}_{\text{world}\rightarrow\text{body}} \tag{4.68}$$

Eq. (4.68) is known as the **quaternion kinematic equation**[11]. Note that we use Newton's notation for the time derivative since $\mathbf{q}$ is just a column vector and $\mathbf{\Omega}$ is just a standard mathematical matrix. Performing the matrix-vector multiplication on the RHS of Eq. (4.68), we get

$$
\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} -pq_1 - qq_2 - rq_3 \\ pq_0 - qq_3 + rq_2 \\ pq_3 + qq_0 - rq_1 \\ -pq_2 + qq_1 + rq_0 \end{bmatrix}
\tag{4.69}
$$

See B.4 for a computational derivation of Eq. (4.69).

Algorithm 47: `quateqn`
Quaternion kinematic equation.

Inputs:
- $\mathbf{q}_{\text{world}\rightarrow\text{body}} \in \mathbb{H}$ - unit quaternion representing the passive rotation from the world frame to the body frame
- $\left[^{\text{body/world}}\boldsymbol{\omega} \right]_{\text{body}}$ - angular velocity of the body frame with respect to the world frame, expressed in the body frame $[\text{rad/s}]$

Procedure:
1. Normalize the quaternion (Algorithm 36).

$$
\mathbf{q}_{\text{world}\rightarrow\text{body}} = \texttt{quatnormalize}(\mathbf{q}_{\text{world}\rightarrow\text{body}})
$$

2. Compute the quaternion derivative, $d(\mathbf{q}_{\text{world}\rightarrow\text{body}})/dt$, where $\mathbf{q}_{\text{world}\rightarrow\text{body}} = (q_0, q_1, q_2, q_3)^T$ and $\left[^{\text{body/world}}\boldsymbol{\omega} \right]_{\text{body}} = (p, q, r)^T$.

$$
\frac{d(\mathbf{q}_{\text{world}\rightarrow\text{body}})}{dt} = \frac{1}{2} \begin{bmatrix} -pq_1 - qq_2 - rq_3 \\ pq_0 - qq_3 + rq_2 \\ pq_3 + qq_0 - rq_1 \\ -pq_2 + qq_1 + rq_0 \end{bmatrix}
$$

3. Return the result.

$$
\textbf{return } \frac{d(\mathbf{q}_{\text{world}\rightarrow\text{body}})}{dt}
$$

Outputs:
- $\dfrac{d(\mathbf{q}_{\text{world}\rightarrow\text{body}})}{dt} \in \mathbb{H}$ - derivative of the unit quaternion from the world frame to the body frame

Note:
- This algorithm assumes the scalar-first convention for quaternions.
- This algorithm normalizes the input quaternion.
- $\mathbf{q}$ is normalized.

Test Cases:
- See Appendix A.4.

---

[11] This equation is derived in various levels of detail in [105, pp. 511–512], [95, pp. 37–38], and [90, pp. 45–46].

4.7.1 Exact Solution for Constant Body Rates

TODO

TODO: attitude error, url: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6864719/

PART II

Dynamics

5

Dynamics

5.1 Bodies, Point Masses, Particles, Forces, and Momenta

A **body** is a physical object that has finite **mass** and **volume** (which is allowed to be infinitesimally small). A **point mass** is an infinitesimally small body with 0 volume (and by extension 0 linear dimensions), thus forming a point [69]. A **particle** is a point mass that is defined to have a constant mass.

A **force** is an external influence acting on a body, altering its motion. From a classical viewpoint, a body's mass is a measure of its resistance to external forces changing its motion. The "motion" we refer to here are its translational kinematics, specifically its inertial velocity.

Momentum, $\boldsymbol{p}$, is the product of a point mass's **mass**, m, and inertial velocity, $^{\mathcal{I}}\boldsymbol{v}$.

$$\boxed{\boldsymbol{p} = m\,^{\mathcal{I}}\boldsymbol{v}} \tag{5.1}$$

It is critical to stress that momentum is a quantity that is only defined relative to an inertial frame (hence why we define it in terms of an inertial velocity) [59].

Angular momentum, $^{O}\boldsymbol{H}$, is the moment of a point mass's momentum about a specific point, O. Let $\boldsymbol{r}_{\cdot/O}$ be the position of the point mass with respect to the point O. Then

$$\boxed{^{O}\boldsymbol{H} = \boldsymbol{r}_{\cdot/O} \times m\,^{\mathcal{I}}\boldsymbol{v}} \tag{5.2}$$

5.2 Newton's Laws of Motion

Newton's laws of motion describe the effects of a net external force acting on a point mass.

5.2.1 First Law: Conservation of Momentum

Newton's First Law: *The momentum of a point mass does not change unless acted upon by a nonzero net external force.*

Mathematically, we can write Newton's first law as

$$\boxed{\left.\frac{d\boldsymbol{p}}{dt}\right|_{\mathcal{I}} = \boldsymbol{0} \qquad \text{(no net force acting on particle)}} \tag{5.3}$$

An alternative way to write this is

$$\boxed{\boldsymbol{p}(t_1) = \boldsymbol{p}(t_2) \qquad \text{(no net force acting on particle)}} \tag{5.4}$$

where $\boldsymbol{p}(t_1)$ and $\boldsymbol{p}(t_2)$ are the momenta of the particle at times t_1 and t_2, respectively [63].

5.2.2 Second Law: Change of Momentum

> **Newton's Second Law:** *The rate of change of a point mass's momentum is equal to the net force applied to the point mass.*

Let $\boldsymbol{F}$ represent the **net force** acting on a point mass. Mathematically, we can write Newton's second law as Eq. (5.5) below. Note that instead of taking a simple time derivative, we take a *frame* derivative (see Section 3.7) since momentum is defined relative to an inertial frame.

$$\boxed{\boldsymbol{F} = \left.\frac{d\boldsymbol{p}}{dt}\right|_{\mathcal{I}}} \tag{5.5}$$

In the special case of a point mass with constant mass, we can simplify Eq. (5.5) to a familiar form.

$$\boldsymbol{F} = \left.\frac{d\boldsymbol{p}}{dt}\right|_{\mathcal{I}} = \left.\frac{d}{dt}\right|_{\mathcal{I}} \left(m \, {}^{\mathcal{I}}\boldsymbol{v}\right) = m \left.\frac{d \, {}^{\mathcal{I}}\boldsymbol{v}}{dt}\right|_{\mathcal{I}}$$

$$\boxed{\boldsymbol{F} = m \, {}^{\mathcal{I}}\boldsymbol{a} \qquad \text{(for a point mass with constant mass)}} \tag{5.6}$$

Note that ${}^{\mathcal{I}}\boldsymbol{a}$ is the inertial acceleration; see Chapter 3 for more details.

5.2.3 Third Law: Reciprocal Forces

> **Newton's Third Law:** *For every action, there is an equal and opposite reaction.*

By "action", Newton's third law implies "force". Consider particle 1 applying a force on particle 2. We denote this force as $\boldsymbol{F}_{1\to2}$. Newton's third law says that there is an equal (in magnitude) and opposite (in direction) force applied by particle 2 on particle 1. Denoting this second force as $\boldsymbol{F}_{2\to1}$, we can write Newton's third law as

$$\boxed{\boldsymbol{F}_{1\to2} = -\boldsymbol{F}_{2\to1}} \tag{5.7}$$

5.2.4 Zeroeth Law: Implicit Laws

In some regards, Newton's laws of motion are incomplete; there are additional "laws" that are typically not explicitly stated but rather implicitly used/applied. We group these additional laws into a single and name it **Newton's zeroeth law**.

> **Newton's Zeroeth Law:**
> 1. The total mass of a two smaller bodies is the sum of their individual masses [63].
> 2. At any instant in time, a body reacts to the forces applied to it at that instant in time [63].
> 3. Forces obey the superposition principle [63].
> 4. Forces change only a body's acceleration; they *cannot* instantaneously change a body's position and velocity.

Item 1

For n bodies, we can write the total mass, m, as the sum of each individual mass, m_i.

$$m = \sum_{i=1}^{n} m_i$$

Item 2

The fact that body's react to forces applied to at that instant in time has already been *implicitly* stated by Eq. (5.5); to be extra explicit regarding this fact, we simply need to add the time argument.

$$\boldsymbol{F}(t) = \left.\frac{d\boldsymbol{p}(t)}{dt}\right|_{\mathcal{I}}$$

In the equation above, $\boldsymbol{p}(t) = m(t)\,^{\mathcal{I}}\boldsymbol{v}(t)$. In the case that the mass is constant (i.e. $m(t) = m = $ constant), we can use Eq. (**??**).

$$\boldsymbol{F}(t) = m\,^{\mathcal{I}}\boldsymbol{a}(t) \qquad \text{(for a point mass with constant mass)}$$

Item 3

The fact that forces obey the superposition can be expressed mathematically as

$$\boldsymbol{F} = \sum_{i=1}^{n} \boldsymbol{F}_i$$

where $\boldsymbol{F}$ is the total force, and $\boldsymbol{F}_i$ is the ith force out of n individual forces applied to the body. This can also be expressed in a coordinate vector form if each force is expressed in the same coordinate frame. For the equation below, we simply use the generic inertial frame $\mathcal{I}$, but *any* coordinate frame (not limited to inertial frames) could be used.

$$[\boldsymbol{F}]_{\mathcal{I}} = \sum_{i=1}^{n} [\boldsymbol{F}_i]_{\mathcal{I}} \in \mathbb{R}^3$$

Item 4

Finally, forces can only change the acceleration of a body instantaneously; they cannot change its position or velocity. While this is implicitly stated through Newton's second law, it is important to stress this point, as it is frequently applied but rarely discussed. As an example, if your velocity changed instantaneously, then your change in momentum would be infinite since you changed velocity without a change in time (i.e. dividing by 0), which would imply you experienced a force with infinite magnitude. Similarly, your position cannot change instantaneously, as that is precisely the definition of teleportation.

5.3   Newton's Laws Applied to Systems of Particles

TODO: REWRITE MOMENTS ABOUT ARBITRARY POINT B TODO: assume system of particles is rigid

Consider a system of n particles. The ith particle has mass m_i and inertial velocity $^{\mathcal{I}}\boldsymbol{v}$. There are two types of forces acting on each particle in the system:

1. **Internal forces:** forces between particles in the system.
2. **External forces:** forces originating from outside the system that are acting on the particles in the system.

Let $\boldsymbol{F}_i$ be the net *external* force acting on particle i, and let $\boldsymbol{F}_{ij}$ be the *internal* force exerted by particle j on particle i. Also note that

$$\boldsymbol{F}_{ii} = 0$$

since a particle cannot exert a force on itself. In general, we assume each particle applies an internal force to every other particle in the system (except for itself).

TODO insert fig

When considering a single particle, the internal forces of the *system* are still external forces on an individual *particle*. For particle i, Newton's second law gives

$$\boldsymbol{F}_i + \sum_{j=1}^{n} \boldsymbol{F}_{ij} = m_i {}^{\mathcal{I}}\boldsymbol{a}_i \tag{5.8}$$

Let $\boldsymbol{r}_i$ be the position of particle i. We can also take the moment of both sides of the above equation about the origin.

$$\boldsymbol{r}_i \times \boldsymbol{F}_i + \sum_{j=1}^{n} (\boldsymbol{r}_i \times \boldsymbol{F}_{ij}) = \boldsymbol{r}_i \times m_i {}^{\mathcal{I}}\boldsymbol{a}_i \tag{5.9}$$

Considering the entire system simultaneously, we can take the summation of Eqs. (5.8) and (5.9) from $i = 1$ to $i = n$.

$$\sum_{i=1}^{n} \boldsymbol{F}_i + \sum_{i=1}^{n}\sum_{j=1}^{n} \boldsymbol{F}_{ij} = \sum_{i=1}^{n} m_i {}^{\mathcal{I}}\boldsymbol{a}_i \tag{5.10}$$

$$\sum_{i=1}^{n} (\boldsymbol{r}_i \times \boldsymbol{F}_i) + \sum_{i=1}^{n}\sum_{j=1}^{n} (\boldsymbol{r}_i \times \boldsymbol{F}_{ij}) = \sum_{i=1}^{n} \left(\boldsymbol{r}_i \times m_i {}^{\mathcal{I}}\boldsymbol{a}_i\right) \tag{5.11}$$

All the internal forces occur in pairs $(\boldsymbol{F}_{ij}, \boldsymbol{f}_{ji})$. For a given pair of internal forces, Newton's third law gives

$$\boldsymbol{F}_{ij} = -\boldsymbol{f}_{ji}$$

so

$$\boldsymbol{F}_{ij} + \boldsymbol{f}_{ji} = \boldsymbol{0} \tag{5.12}$$

and by extension

$$\underbrace{(\boldsymbol{f}_{12} + \boldsymbol{f}_{21})}_{=\boldsymbol{0}} + \underbrace{(\boldsymbol{f}_{13} + \boldsymbol{f}_{31})}_{=\boldsymbol{0}} + \cdots = \sum_{i=1}^{n}\sum_{j=1}^{n} \boldsymbol{F}_{ij} = \boldsymbol{0}$$

Substituting this result into Eq. (5.10) [83, pp. 176–177], [14, pp. 868–870],

$$\boxed{\sum_{i=1}^{n} \boldsymbol{F}_i = \sum_{i=1}^{n} m_i {}^{\mathcal{I}}\boldsymbol{a}_i} \tag{5.13}$$

As for the moments, we can show that the sum of the moments of a pair of internal forces evaluates to 0. Let's begin by considering the sum

$$\boldsymbol{r}_i \times \boldsymbol{F}_{ij} + \boldsymbol{r}_j \times \boldsymbol{f}_{ji}$$

We can add the term $(\boldsymbol{r}_i \times \boldsymbol{f}_{ji}) - (\boldsymbol{r}_i \times \boldsymbol{f}_{ji})$ since it is just equal to $\boldsymbol{0}$.

$$[\boldsymbol{r}_i \times \boldsymbol{F}_{ij} + \boldsymbol{r}_j \times \boldsymbol{f}_{ji}] = [\boldsymbol{r}_i \times \boldsymbol{F}_{ij} + \boldsymbol{r}_j \times \boldsymbol{f}_{ji}] + (\boldsymbol{r}_i \times \boldsymbol{f}_{ji}) - (\boldsymbol{r}_i \times \boldsymbol{f}_{ji})$$

Rearranging and grouping terms,

$$[\boldsymbol{r}_i \times \boldsymbol{F}_{ij} + \boldsymbol{r}_j \times \boldsymbol{f}_{ji}] = \boldsymbol{r}_i \times (\boldsymbol{F}_{ij} + \boldsymbol{f}_{ji}) + (\boldsymbol{r}_j - \boldsymbol{r}_i) \times \boldsymbol{f}_{ji}$$

Substituting Eqs. (5.12) (i.e. Newton's third law),

$$[\boldsymbol{r}_i \times \boldsymbol{F}_{ij} + \boldsymbol{r}_j \times \boldsymbol{f}_{ji}] = (\boldsymbol{r}_j - \boldsymbol{r}_i) \times \boldsymbol{f}_{ji}$$

Finally, we know that $(\boldsymbol{r}_j - \boldsymbol{r}_i)$ and $\boldsymbol{f}_{ji}$ are collinear, since $(\boldsymbol{r}_j - \boldsymbol{r}_i)$ is the position of particle j with respect to particle i, and the internal force between those particles must be along that relative position vector. Since those vectors are collinear, their cross product is 0, and we have

$$[\boldsymbol{r}_i \times \boldsymbol{F}_{ij} + \boldsymbol{r}_j \times \boldsymbol{f}_{ji}] = \boldsymbol{0}$$

Taking the summation over all pairs of particles,

$$\sum_{i=0}^{n}\sum_{j=0}^{n}(\boldsymbol{r}_i \times \boldsymbol{F}_{ij}) = \sum_{i=0}^{n}\sum_{j=0}^{n}\boldsymbol{0} \quad \rightarrow \quad \sum_{i=0}^{n}\sum_{j=0}^{n}(\boldsymbol{r}_i \times \boldsymbol{F}_{ij}) = \boldsymbol{0}$$

Substituting this result into Eq. (5.11) [83, p. 179], [14, pp. 868–870],

$$\sum_{i=1}^{n}(\boldsymbol{r}_i \times \boldsymbol{F}_i) = \sum_{i=1}^{n}\left(\boldsymbol{r}_i \times m_i\,{}^{\mathcal{I}}\boldsymbol{a}_i\right) \tag{5.14}$$

5.3.1   Center of Mass

The **center of mass** for a system of particles, $\bar{\boldsymbol{r}}$, is defined as

$$\bar{\boldsymbol{r}} = \frac{1}{m}\sum_{i=1}^{n}m_i\boldsymbol{r}_i \tag{5.15}$$

where

$$m = \sum_{i=1}^{n}m_i \tag{5.16}$$

is the **total system mass** [14, pp. 872–873], [83, p. 177].

5.3.2   Momentum and Newton's Second Law for Systems

The momentum of the ith particle is simply

$$\boldsymbol{p}_i = m_i\,{}^{\mathcal{I}}\boldsymbol{v}_i$$

The **total system momentum**, $\boldsymbol{p}_{\text{sys}}$, is then

$$\boldsymbol{p}_{\text{sys}} = \sum_{i=1}^{n}\boldsymbol{p}_i = \sum_{i=1}^{n}m_i\,{}^{\mathcal{I}}\boldsymbol{v}_i \tag{5.17}$$

Let's rearrange Eq. (5.15).

$$m\bar{\boldsymbol{r}} = \sum_{i=1}^{n}m_i\boldsymbol{r}_i$$

Taking the frame derivative[1] of both sides of Eq. (5.19) with respect to the inertial frame

$$\left.\frac{d}{dt}\right|_{\mathcal{I}} m\bar{\boldsymbol{r}} = \left.\frac{d}{dt}\right|_{\mathcal{I}} \sum_{i=1}^{n}m_i\boldsymbol{r}_i$$

---

[1]   See Section 3.7.

Noting that the mass of each individual particle remains constant (and by extension the total system mass also remains constant),

$$m\frac{d\bar{\boldsymbol{r}}}{dt}\bigg|_{\mathcal{I}} = \sum_{i=1}^{n} m_i \frac{d\boldsymbol{r}_i}{dt}\bigg|_{\mathcal{I}}$$

$$m\,^{\mathcal{I}}\bar{\boldsymbol{v}} = \sum_{i=1}^{n} m_i\,^{\mathcal{I}}\boldsymbol{v}_i \tag{5.18}$$

Note that $^{\mathcal{I}}\bar{\boldsymbol{v}}$ is the inertial velocity of the system center of mass. By definition, the right hand side of the Eq. (5.18) is the total system momentum. Substituting Eq. (5.18) into Eq. (5.17),

$$\boxed{\boldsymbol{p}_{\text{sys}} = m\,^{\mathcal{I}}\bar{\boldsymbol{v}}} \tag{5.19}$$

Now, let's take the frame derivative[2] of both sides of Eq. (5.19) with respect to the inertial frame,

$$\frac{d\boldsymbol{p}_{\text{sys}}}{dt}\bigg|_{\mathcal{I}} = \frac{d}{dt}\bigg|_{\mathcal{I}} \sum_{i=1}^{n} \boldsymbol{p}_i = \sum_{i=1}^{n} \frac{d\boldsymbol{p}_i}{dt}\bigg|_{\mathcal{I}} = \sum_{i=1}^{n} \frac{d}{dt}\bigg|_{\mathcal{I}} \left(m_i\,^{\mathcal{I}}\boldsymbol{v}_i\right)$$

Since the mass of an individual particle is constant (see Section 5.1),

$$\frac{d\boldsymbol{p}_{\text{sys}}}{dt}\bigg|_{\mathcal{I}} = \sum_{i=1}^{n} m_i \frac{d\left(^{\mathcal{I}}\boldsymbol{v}_i\right)}{dt}\bigg|_{\mathcal{I}} = \sum_{i=1}^{n} m_i\,^{\mathcal{I}}\boldsymbol{a}_i$$

However, from Eq. (5.10), we know that

$$\sum_{i=1}^{n} \boldsymbol{F}_i = \sum_{i=1}^{n} m_i\,^{\mathcal{I}}\boldsymbol{a}_i$$

Substituting this into the equation above, we get

$$\frac{d\boldsymbol{p}_{\text{sys}}}{dt}\bigg|_{\mathcal{I}} = \sum_{i=1}^{n} \boldsymbol{F}_i$$

Finally, let's define the **total system external force**, $\boldsymbol{F}_{\text{sys}}$, as

$$\boxed{\boldsymbol{F}_{\text{sys}} = \sum_{i=1}^{n} \boldsymbol{F}_i} \tag{5.20}$$

Then we get

$$\frac{d\boldsymbol{p}_{\text{sys}}}{dt}\bigg|_{\mathcal{I}} = \boldsymbol{F}_{\text{sys}} \tag{5.21}$$

Finally, we can also take the frame derivative of Eq. (5.19) with respect to the inertial frame.

$$\frac{d\boldsymbol{p}_{\text{sys}}}{dt}\bigg|_{\mathcal{I}} = \frac{d}{dt}\bigg|_{\mathcal{I}} \left(m\,^{\mathcal{I}}\bar{\boldsymbol{v}}\right)$$

Again, noting that the total system mass, m, remains constant,

$$\frac{d\boldsymbol{p}_{\text{sys}}}{dt}\bigg|_{\mathcal{I}} = m\frac{d\left(^{\mathcal{I}}\bar{\boldsymbol{v}}\right)}{dt}\bigg|_{\mathcal{I}} = m\,^{\mathcal{I}}\bar{\boldsymbol{a}}$$

where $^{\mathcal{I}}\bar{\boldsymbol{a}}$ is the inertial acceleration of the system center of mass. Combining this expression with Eq. (5.21) [14, pp. 871–874], [83, pp. 176–178]

$$\boxed{\frac{d\boldsymbol{p}_{\text{sys}}}{dt}\bigg|_{\mathcal{I}} = \boldsymbol{F}_{\text{sys}} = m\,^{\mathcal{I}}\bar{\boldsymbol{a}}} \tag{5.22}$$

---

[2]   See Section 3.7.

5.3.3 Angular Momentum and Newton's Second Law for Rotation

Recall from Section 5.1 that the angular momentum of a particle about a point B is defined as

$$^B\boldsymbol{H} = \boldsymbol{r}_{\cdot/B} \times m\,^{\mathcal{I}}\boldsymbol{v}$$

Let $\boldsymbol{r}_{i/B}$ be the position of particle P_i with respect to the point B. Then the angular momentum of particle P_i about the point B is

$$^B\boldsymbol{H}_i = \boldsymbol{r}_{i/B} \times m_i\,^{\mathcal{I}}\boldsymbol{v}_i$$

The **total system angular momentum** about the point B is then

$$^B\boldsymbol{H}_{\text{sys}} = \sum_{i=1}^{n} \left(\boldsymbol{r}_{i/B} \times m_i\,^{\mathcal{I}}\boldsymbol{v}_i \right)$$

Taking the frame derivative of both sides with respect to the inertial frame,

$$\frac{d\left(^B\boldsymbol{H}_{\text{sys}}\right)}{dt}\bigg|_{\mathcal{I}} = \frac{d}{dt}\bigg|_{\mathcal{I}} \left[\sum_{i=1}^{n} \left(\boldsymbol{r}_{i/B} \times m_i\,^{\mathcal{I}}\boldsymbol{v}_i \right) \right] = \sum_{i=1}^{n} \frac{d}{dt}\bigg|_{\mathcal{I}} \left(\boldsymbol{r}_{i/B} \times m_i\,^{\mathcal{I}}\boldsymbol{v}_i \right)$$

$$= \sum_{i=1}^{n} \left(\frac{d\boldsymbol{r}_{i/B}}{dt}\bigg|_{\mathcal{I}} \times m_i\,^{\mathcal{I}}\boldsymbol{v}_i \right) + \sum_{i=1}^{n} \left(\boldsymbol{r}_{i/B} \times m_i \frac{d\left(^{\mathcal{I}}\boldsymbol{v}_i\right)}{dt}\bigg|_{\mathcal{I}} \right)$$

$$= \sum_{i=1}^{n} \left(^{\mathcal{I}}\boldsymbol{v}_{i/B} \times m_i \right)$$

Since the system of particles is rigid, $\boldsymbol{r}$

PART III
Orbital Mechanics

6

Fundamentals of Orbital Mechanics

TODO: update all tikz diagrams in separate latex doc

6.1 Inertial Frames and Notation

This chapter introduces some of the fundamental physical and kinematic relationships that form the basis for much of orbital mechanics. **Most notably, these fundamentals always assume that we are dealing with kinematics in an inertial frame.** As such, we simplify our notation in comparison to the previous chapters, dropping the frame labels from variables and writing time derivatives using Newton's notations. For example, we could write the inertial acceleration simply as $\boldsymbol{a}$ instead of $^I\boldsymbol{a}$, and also write it as the second derivative of position using Newton's notation as $\boldsymbol{a} = \ddot{\boldsymbol{r}}$, or as the first derivative of velocity, $\boldsymbol{a} = \dot{\boldsymbol{v}}$.

> Most introductory texts on orbital mechanics use the same style of notation for these relationships (i.e. Newton's notation for derivatives + no frame labels). For more more information on these notation simplifications for the special case of kinematics in inertial frames, see Section **??**.

Convention 38: Simplified notation for orbital mechanics.

Vectors relative to an inertial frame follow the simplified notation from Section **??**, where:

1. The frame label is neglected.
2. Newton's notation can be used for time derivatives.

For example, the the inertial velocity and its magnitude is just

$$\boldsymbol{v} = \dot{\boldsymbol{r}}$$
$$v = \dot{r}$$

while the inertial acceleration and its magnitude is

$$\boldsymbol{a} = \dot{\boldsymbol{v}} = \ddot{\boldsymbol{r}}$$
$$a = \dot{v} = \ddot{r}$$

In some cases, we will continue adding the inertial frame label $\mathcal{I}$ to help differentiate from other frames. However, if the frame label is not included, it is safe to assume that the vector is relative to an inertial frame.

Classically[1], most introductory texts on orbital mechanics also use the simplified notation outlined by Convention 38. While it is easy to add the frame label to vectors, the majority of the fundamental relationships in orbital mechanics are scalar equations involving the magnitudes of vectors relative to an inertial frame. Including the inertial frame label, $\mathcal{I}$, would clutter these equations, and make them look significantly different from their "standard" form that you'd find in other resources.

6.2   Newton's Law of Universal Gravitation

Consider two bodies, 1 and 2, that have mass m_1 and m_2, respectively. The two bodies have positions $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ in an inertial coordinate frame, $\mathcal{I}$. The relative position of body 2 with respect to body 1 is simply

$$\boxed{\boldsymbol{r}_{2/1} = \boldsymbol{r}_2 - \boldsymbol{r}_1}$$ (6.1)
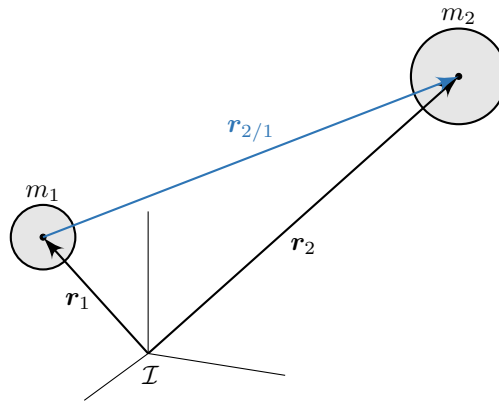
Figure 6.1: Relative positions of bodies.

Both masses produce a gravitational field, and both experience a gravitational force due to the other's gravitational field. The notation we use throughout this chapter for the gravitational force is defined by Convention 39 below.

Convention 39: Notation for the gravitational force between two bodies.

$$\boldsymbol{F}_{kj} = \text{gravitational force exerted on body } k \text{ by the gravitational field of body } j$$

---

[1]   One notable exception is the book by Rao [75], which *does* include frame labels on vectors but *not* on scalars. However, this is somewhat of an inconsistent notation, since it is essentially using one convention for vectors (i.e. including frame labels) and another for scalar (i.e. not including frame labels for inertial vectors).

Newton's law of universal gravitation gives

$$\boldsymbol{F}_{21} = -\frac{Gm_1m_2}{r_{2/1}^2}\hat{\mathbf{r}}_{2/1} \tag{6.2}$$

where G is the **universal gravitational constant** and where $r_{2/1} = \left|\boldsymbol{r}_{2/1}\right|$. The value of G is only known to a very limited accuracy, and values tend to differ slightly from resource to resource. NIST provides [64]

$$\boxed{G = 6.67430 \times 10^{-11} \pm 1.5 \times 10^{-6} \text{ m}^3/(\text{kg} \cdot \text{s}^2)} \tag{6.3}$$

We can write the unit vector in Eq. (6.2) as

$$\hat{\mathbf{r}}_{2/1} = \frac{\boldsymbol{r}_{2/1}}{\left|\boldsymbol{r}_{2/1}\right|} = \frac{\boldsymbol{r}_{2/1}}{r_{2/1}}$$

so we can rewrite Eq. (6.2) as

$$\boxed{\boldsymbol{F}_{21} = -\frac{Gm_1m_2}{r_{2/1}^3}\boldsymbol{r}_{2/1}} \tag{6.4}$$

By Eq. (6.4), we also know that the gravitational force exerted on body 1 by the gravitational field of body 2 is

$$\boldsymbol{F}_{12} = -\frac{Gm_1m_2}{r_{1/2}^3}\boldsymbol{r}_{1/2}$$

We know that $\boldsymbol{r}_{1/2} = -\boldsymbol{r}_{2/1}$, and $r_{1/2} = r_{2/1}$. Thus,

$$\boldsymbol{F}_{12} = \frac{Gm_1m_2}{r_{2/1}^3}\boldsymbol{r}_{2/1}$$

and we have that

$$\boxed{\boldsymbol{F}_{12} = -\boldsymbol{F}_{21}} \tag{6.5}$$

This paired gravitational interaction is depicted in Fig. 6.2 below [65].
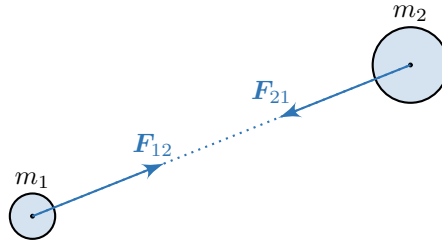
Figure 6.2: Gravitational interaction between two bodies.

6.2.1   Assumptions

Newton's law of universal gravitation assumes the following:

1. The bodies are spherical.
2. The bodies have a uniform density.

These two assumptions result in the inverse square law form of the gravitational force given by Newton's law of universal gravitation. While this assumption is not true in general, it provides a decent approximation for the case of small satellites orbiting large celestial bodies, and facilitates the development of very simple yet powerful orbital mechanics concepts.

Chapter 11 details a spherical harmonic gravity model that is used for high-fidelity simulations. Additionally, a particular truncation of that model allows for simple modifications to Keplerian orbits that are useful for both lower fidelity orbit propagation and for orbit design.

6.3   The N-Body Problem

Consider N bodies (of positive, nonzero mass) in an inertial coordinate frame. Each body will have gravitational interactions with every other $N-1$ bodies. In Fig. 6.3, we only illustrate a single pair of these gravitational interactions, but these interactions occur along every dotted line in that figure. Applying Newton's law of universal gravitation to
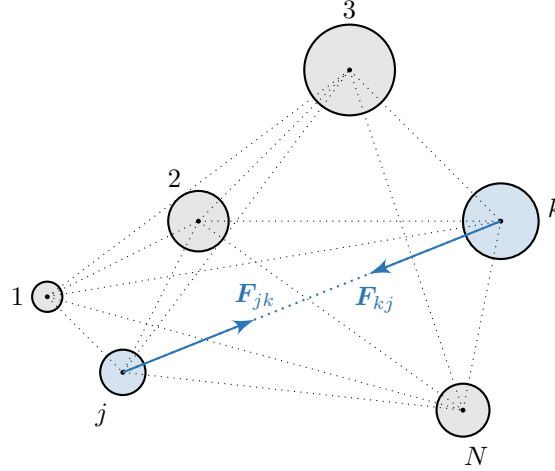
Figure 6.3: The N-body problem.

find the force acting on the jth body due to the gravitational field of the kth body,

$$\boldsymbol{F}_{jk} = -\frac{Gm_k m_j}{r_{j/k}^3}\boldsymbol{r}_{j/k}$$

The total gravitational force exerted on the jth body by the other $N-1$ bodies is then

$$\boldsymbol{F}_j = \sum_{\substack{k=1 \\ k\neq j}}^{N} \boldsymbol{F}_{jk} = \sum_{\substack{k=1 \\ k\neq j}}^{N} \left[-\frac{Gm_k m_j}{r_{j/k}^3}\boldsymbol{r}_{j/k} \right] = -Gm_j \sum_{\substack{k=1 \\ k\neq j}}^{N} m_k \frac{\boldsymbol{r}_{j/k}}{r_{j/k}^3} \tag{6.6}$$

> **Convention 40:** Total gravitational force acting on a body.
>
> $\boldsymbol{F}_j$ = gravitational force exerted on body j by the gravitational fields of the other $N-1$ bodies

From Newton's second law (for constant mass), we know

$$\boldsymbol{F}_j = m_j \boldsymbol{a}_j \tag{6.7}$$

where

$$\boldsymbol{a}_j = \ddot{\boldsymbol{r}}_j$$

is the inertial[2] acceleration of body j. Equating Eqs. (6.6) and (6.7) and substituting the inertial acceleration with the second derivative of position,

$$m_j \ddot{\boldsymbol{r}}_j = -Gm_j \sum_{\substack{k=1 \\ k\neq j}}^{N} m_k \frac{\boldsymbol{r}_{j/k}}{r_{j/k}^3}$$

---

[2]   See the introduction to this Chapter, as well as Section **??**, for this simplified notation.

$$\ddot{\boldsymbol{r}}_j = -G \sum_{\substack{k=1 \\ k \neq j}}^{N} m_k \frac{\boldsymbol{r}_{j/k}}{r_{j/k}^3} \tag{6.8}$$

Since there are N bodies, we have a system of N nonlinear, 2nd-order, ordinary differential equations, where the jth equation takes the form of Eq. (6.9).

$$
\begin{aligned}
\ddot{\boldsymbol{r}}_1 &= -G \sum_{k=2}^{N} m_k \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \\
&\vdots \\
\ddot{\boldsymbol{r}}_j &= -G \sum_{\substack{k=1 \\ k \neq j}}^{N} m_k \frac{\boldsymbol{r}_{j/k}}{r_{j/k}^3} \\
&\vdots \\
\ddot{\boldsymbol{r}}_N &= -G \sum_{k=1}^{N-1} m_k \frac{\boldsymbol{r}_{N/k}}{r_{N/k}^3}
\end{aligned}
\tag{6.9}
$$

Eq. (6.9) defines the **N-body problem**. In general, the N-body problem is chaotic; small perturbations in its initial conditions result in drastically different trajectories. From a numerical standpoint, this also means that any small numerical error will increase exponentially as the numerical integration progresses [1], [13, pp. 5–7]. There are two cases where a general closed-form solution exists:

1. $N = 1$: trivial solution (a single mass not experiencing gravitation from another mass will not accelerate with respect to inertial space)
2. $N = 2$: two-body problem – this solution is covered in detail in the next chapter

6.3.1   A Convenient Reformulation

From the N-body problem, we have

$$\ddot{\boldsymbol{r}}_j = -G \sum_{\substack{k=1 \\ k \neq j}}^{N} m_k \frac{\boldsymbol{r}_{j/k}}{r_{j/k}^3}$$

Let's consider the motion of body 2 relative to body 1. From the N-body problem, for body 1, we can write

$$
\begin{aligned}
\ddot{\boldsymbol{r}}_1 &= -G \sum_{\substack{k=1 \\ k \neq 1}}^{N} m_k \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \\
&= -G \sum_{k=2}^{N} m_k \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3}
\end{aligned}
$$

$$\ddot{\boldsymbol{r}}_1 = -G m_2 \frac{\boldsymbol{r}_{1/2}}{r_{1/2}^3} - G \sum_{k=3}^{N} m_k \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \tag{6.10}$$

Similarly, for body 2, we can write

$$\ddot{\boldsymbol{r}}_2 = -G \sum_{\substack{k=1 \\ k \neq 2}}^{N} m_k \frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3}$$

$$\ddot{\boldsymbol{r}}_2 = -G m_1 \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} - G \sum_{k=3}^{N} m_k \frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} \tag{6.11}$$

We know

$$\boldsymbol{r}_{2/1} = \boldsymbol{r}_2 - \boldsymbol{r}_1$$

Differentiating twice in time (relative to the inertial frame),

$$\dot{\boldsymbol{r}}_{2/1} = \dot{\boldsymbol{r}}_2 - \dot{\boldsymbol{r}}_1$$

$$\ddot{\boldsymbol{r}}_{2/1} = \ddot{\boldsymbol{r}}_2 - \ddot{\boldsymbol{r}}_1 \tag{6.12}$$

Substituting Eqs. (6.10) and (6.11) into Eq. (6.12),

$$\ddot{\boldsymbol{r}}_{2/1} = \left[-Gm_1 \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} - G\sum_{k=3}^{N} m_k \frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} \right] - \left[-Gm_2 \frac{\boldsymbol{r}_{1/2}}{r_{1/2}^3} - G\sum_{k=3}^{N} m_k \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right]$$

$$= -Gm_1 \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} + Gm_2 \frac{\boldsymbol{r}_{1/2}}{r_{1/2}^3} - G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right)$$

Since $\boldsymbol{r}_{1/2} = -\boldsymbol{r}_{2/1}$,

$$\ddot{\boldsymbol{r}}_{2/1} = -Gm_1 \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} - Gm_2 \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} - G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right)$$

$$\boxed{\ddot{\boldsymbol{r}}_{2/1} = -G(m_1 + m_2) \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} - G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right)} \tag{6.13}$$

This formulation is useful for immediately extracting the two-body problem (see Section **??**), formulating an expression for a general gravitational perturbing acceleration (see Section **??**), and for formulating an expression for third-body gravitational perturbations (see Section **??**) [13, pp. 7–11].

6.4 The Two-Body Problem

The **two-body problem** considers only two interacting masses, i.e. $N = 2$. However, we can make **either of** the following assumptions to obtain the two-body problem directly from the N-body problem:

- **Assumption #1:** Each m_k is small in comparison to m_1.

    OR

- **Assumption #2:** Each body is far away from m_1 and m_2.

To see the effects of these assumptions, recall Eq. (**??**), repeated below for convenience.

$$\ddot{\boldsymbol{r}}_{2/1} = -G(m_1 + m_2) \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} - G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right) \tag{??}$$

If we make assumption #1 (i.e. $m_1, m_2 \gg m_k$ for all k), then the first term dominates the second term.

$$\left[G(m_1 + m_2) \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} \right] \gg \left[G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right) \right]$$

$$\therefore \ddot{\boldsymbol{r}}_{2/1} \approx -G(m_1 + m_2) \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3}$$

Alternatively, we can make assumption #2, which assumes that each body is far away from m_1 and m_2. Mathematically,

$$r_{1/k} \gg r_{2/1}$$
$$r_{2/k} \gg r_{2/1}$$

This implies that[3]

$$\boldsymbol{r}_{1/k} \approx \boldsymbol{r}_{2/k}$$
$$\boldsymbol{r}_{1/k} \approx \boldsymbol{r}_{2/k}$$

and the second term in Eq. (**??**) will be approximately 0.

$$\ddot{\boldsymbol{r}}_{2/1} = -G(m_1 + m_2)\frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} - \underbrace{G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right)}_{\approx 0 \text{ if } \boldsymbol{r}_{1/k} \approx \boldsymbol{r}_{2/k}}$$

$$\therefore \ddot{\boldsymbol{r}}_{2/1} \approx -G(m_1 + m_2)\frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3}$$

In either case (i.e. whether we made assumption #1 or #2, we end up with the same equation. This is the same equation that would result if we had only two bodies in the first place, which is

$$\therefore \ddot{\boldsymbol{r}}_{2/1} = -G(m_1 + m_2)\frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3} \tag{6.14}$$

We define the **standard gravitational parameter** (or simply **gravitational parameter**), μ, as

$$\boxed{\mu = G(m_1 + m_2)} \tag{6.15}$$

Then Eq. (6.14) can be written as

$$\boxed{\ddot{\boldsymbol{r}}_{2/1} = -\mu \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3}} \tag{6.16}$$

Equation (6.16) is a formulation of the two-body problem where we can solve for the motion of body 2 with respect to body 1. Another common formulation of the two-body problem describes the motion of either body with respect to their mutual center of mass (i.e. barycenter) [100], [13, pp. 11–14].

6.4.1 Perturbing Acceleration

Recall Eq. (**??**):

$$\ddot{\boldsymbol{r}}_{2/1} = -G(m_1 + m_2)\frac{\boldsymbol{r}_{2/1}}{r_{12}^3} - G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right) \tag{??}$$

We can notice that there are two contributions to the acceleration of body 2 with respect to body 1:

$$\ddot{\boldsymbol{r}}_{2/1} = \underbrace{\left[-G(m_1 + m_2)\frac{\boldsymbol{r}_{2/1}}{r_{12}^3} \right]}_{\text{two-body acceleration}} + \underbrace{\left[-G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right) \right]}_{\text{perturbing acceleration}}$$

We can thus define the **perturbing acceleration**, $\ddot{\boldsymbol{r}}_{\text{pert}}$, as

$$\boxed{\ddot{\boldsymbol{r}}_{\text{pert}} = -G\sum_{k=3}^{N} m_k \left(\frac{\boldsymbol{r}_{2/k}}{r_{2/k}^3} - \frac{\boldsymbol{r}_{1/k}}{r_{1/k}^3} \right)} \tag{6.17}$$

---

[3] Note that we can switch the vector form because if body 1 and 2 are much closer to each other than they are to body k, then the vector connecting bodies 1 and k will be almost collinear with the vector connecting bodies 2 and k.

6.5 The Fundamental Orbital Differential Equation

For the two body problem, we found[4]

$$\ddot{\boldsymbol{r}}_{2/1} = -\mu \frac{\boldsymbol{r}_{2/1}}{r_{2/1}^3}$$

where

$$\mu = G(m_1 + m_2)$$

Now, let us consider the case where we have a satellite of mass m and a central body of mass M. The satellite has position vector $\boldsymbol{r}$ with respect to the central body (i.e. $\boldsymbol{r}$ is the position of the satellite's center of mass relative to the central body's center of mass). If we let the satellite be body 1 and the central body be body 2, then we have

$$\ddot{\boldsymbol{r}} = -\mu \frac{\boldsymbol{r}}{r^3} \tag{6.18}$$

where $\ddot{\boldsymbol{r}}$ is the acceleration of the satellite relative to an inertial frame centered at the center of mass of the central body, and where

$$\mu = G(M + m)$$

is the gravitational parameter. Finally, with the assumption that the mass of the central body is much greater than the mass of the satellite ($M \gg m$), we can write the gravitational parameter as [13, pp. 13–14]

$$\boxed{\mu = GM} \tag{6.19}$$

Finally, rearranging Eq. (6.18) such that all nonzero terms are on the left hand side,

$$\boxed{\ddot{\boldsymbol{r}} + \mu \frac{\boldsymbol{r}}{r^3} = \boldsymbol{0}} \tag{6.20}$$

Equation (6.20) is known as the **fundamental orbital differential equation (FODE)**.

> The assumptions we made to arrive at Eq. (6.20) are as follow:
>
> 1. Both bodies are spherical and have a uniform density (assumed by Newton's law of universal gravitation).
> 2. Both bodies have constant mass.
> 3. All dynamics are relative to an inertial reference frame.
> 4. The gravitational contributions of any other bodies external to the satellite-central body system are negligible.
> 5. The mass of the satellite is negligible in comparison the mass of the central body.
>
> Note that all of the assumptions except for the last one also apply for the slightly more general two-body problem.

6.6 Specific Mechanical Energy

The mechanical energy $E_{\text{mechanical}}$ of an orbiting body is defined as

$$E_{\text{mechanical}} = K + U \tag{6.21}$$

---

[4] See the previous section.

where K is its kinetic energy given by

$$K = \frac{1}{2}mv^2 \tag{6.22}$$

and where U is its gravitational potential energy.

To obtain the gravitational potential energy, consider Newton's law of universal gravitation from Eq. (**??**), repeated below for convenience:

$$\boldsymbol{F}_{21} = -\frac{Gm_1m_2}{r_{2/1}^3}\boldsymbol{r}_{2/1}$$

In the context of the satellite-central body system (see Section **??**), we write

$$\boldsymbol{F} = -\frac{GMm}{r^3}\boldsymbol{r} = -\frac{\mu m}{r^3}\boldsymbol{r}$$

Consider the case where the satellite starts at a radial distance r_1, and the gravitational force exerted on it by the central body brings it to a new radial distance r_2. Since the gravitational force is conservative, we can calculate the work done by gravity in this process, $W_{1\to2}$, as

$$W_{1\to2} = \int_{r_1}^{r_2} \boldsymbol{F} \cdot d\boldsymbol{r}$$

Substituting our force expression into the work expression above,

$$W_{1\to2} = \int_{r_1}^{r_2} \left[-\frac{\mu m}{r^3}\boldsymbol{r}\right] \cdot d\boldsymbol{r} = -\mu m \int_{r_1}^{r_2} \left(\frac{\boldsymbol{r} \cdot d\boldsymbol{r}}{r^3}\right) = -\mu m \int_{r_1}^{r_2} \frac{r}{r^3}\,dr = -\mu m \int_{r_1}^{r_2} \frac{dr}{r^2} = \frac{\mu m}{r}\bigg|_{r_1}^{r_2}$$

$$= \mu m \left(\frac{1}{r_2} - \frac{1}{r_1}\right)$$

The satellite has some potential energy U_1 at r_1 and some other potential energy U_2 at r_2. Its change in gravitational potential energy as it moves from r_1 from r_2 is

$$\Delta U = U_2 - U_1$$

Fundamentally, the change in potential energy in a conservative force field is related to the work done by that conservative force field as

$$\Delta U = -W_{1\to2}$$

Therefore, we have

$$\Delta U = \mu m \left(\frac{1}{r_1} - \frac{1}{r_2}\right) \tag{6.23}$$

Equation (6.23) tells us the *change* in gravitational potential energy. But what is *the* value of the gravitational potential energy of the satellite at an arbitrary distance r from the center of mass of the central body? Since Eq. (6.23) defines a *change* in U, we need to define a reference datum. We could define this datum at an arbitrary location in space. However, for orbital mechanics, we need a common reference datum that is convenient for *all* celestial bodies *anywhere* in the universe [103, p. 27]. This logically leaves us with a single choice for the reference datum:

> **Convention 41:** Reference datum for gravitational potential energy.
>
> The reference datum which we measure gravitational potential energy with respect to is located at $r = \infty$.

Thus, the gravitational potential energy at a location r is defined as the *change* in gravitational potential enery as the satellite moves from ∞ to r. Therefore, in Eq. (6.23), we let $r_2 = r$ and take the limit as $r_1 \to \infty$ [13, pp. 14–16], [55, 108].

$$U = \lim_{r_1 \to \infty} \left[m\mu \left(\frac{1}{r_1} - \frac{1}{r}\right)\right]$$

$$U = -\frac{\mu m}{r} \tag{6.24}$$

The **gravitational potential**, V, of a body is its specific gravitational potential energy (i.e. U per unit mass).

$$V = -\frac{\mu}{r} \tag{6.25}$$

Now, we are ready to develop an expression for the **specific mechanical energy**, $\mathcal{E}$. By definition,

$$\mathcal{E} = \frac{E_{\text{mechanical}}}{m} = \frac{K + U}{m} = \frac{K}{m} + V$$

Substituting our expressions for K and U into the equation above,

$$\mathcal{E} = \frac{v^2}{2} - \frac{\mu}{r} \tag{6.26}$$

6.7 Specific Angular Momentum

Recall the definition of angular momentum:

$$\boldsymbol{H} = \boldsymbol{r} \times m\boldsymbol{v}$$

The specific angular momentum is therefore just [13, p. 17]

$$\boldsymbol{h} = \boldsymbol{r} \times \boldsymbol{v} \tag{6.27}$$

6.8 Astronomical Symbols

In orbital mechanics, we typically label variables associated with a celestial body with a pictorial symbol representing that celestial body.

Table 6.1: Astronomical symbols for celestial bodies.

Celestial Body	Symbol
Sun	☉
Mercury	☿
Venus	♀
Earth	⊕
Moon[5]	☽
Mars	♂
Jupiter	♃
Saturn	♄
Uranus	♅
Neptune	♆
Pluto	♇

Additionally, there are a few other symbols we use in orbital mechanics to help define specific points along our orbit. See Section (TODO: cite section) for more information on these points [13, p. 41], [8, 44].

---

[5] We use a decrescent (waning) Moon to (a) avoid it looking similar to a C in a subscript and (b) because the spacing between the variable and the subscript is too large. For example, $r_{\mathbb{C}}$ vs. $r_{\mathbb{D}}$. [103] and [13, p.41] use this symbol for the Moon.

Table 6.2: Astronomical symbols for orbital reference points.

Reference Point	Symbol
ascending node	☊
descending node	☋
vernal equinox (first point of Aries)	♈

7

Orbits in Two Dimensions

TODO: update diagrams in Tikz TODO: note on continued use of simplified kinematic notation TODO: sphere of influence

Simply put, an **orbit** is the trajectory of a body about a position in space. The orbiting body is referred to as a **satellite**, while the position it is orbiting about is typically a planet, moon, or star. In the case where the mass of the satellite is not insignificant, or if there are multiple large bodies near the satellite, the satellite's orbit is about a barycenter. In this chapter, we focus on the case of a small satellite orbiting a single large body, as defined by the Fundamental Orbital Differential Equation (FODE) from Section **??**. Sections 7.1 and 7.2 provide us with some mathematical tools and concepts for solving and analyzing the FODE, as well as some tools for analyzing orbits in later parts of this chapter and beyond. Sections 7.3 and 7.4 identify constants of motion that will aid us in solving the FODE, and Section **??** introduces the flight path angle, which is closely coupled with the angular momentum of the orbit due to the conservation introduced in Section 7.4. Section **??** is where we actually solve the FODE, and the remainder of the chapter is dedicated to the analysis of this solution in two dimensions. We then extend this solution to three dimensions in the next chapter (Chapter **??**.

TODO: update intro above because FPA is not its own section anymore

7.1 Mathematical Tools

7.1.1 Scalar Derivative Identities

Consider an arbitrary scalar variable, $s \in \mathbb{R}$, that is a function of time, t. Below we derive two identities involving time variables of functions of s that will be useful in later derivations [13, p. 15].

$$\frac{d}{dt}\left(\frac{s^2}{2}\right) = \frac{1}{2}\frac{d}{dt}(s^2) = \frac{1}{2}\left(2s\frac{ds}{dt}\right) = s\dot{s} \quad \rightarrow \quad \boxed{\frac{d}{dt}\left(\frac{s^2}{2}\right) = s\dot{s}} \tag{7.1}$$

$$\frac{d}{dt}\left(\frac{1}{s}\right) = \frac{d}{dt}(s^{-1}) = -s^{-2}\frac{ds}{dt} = -\frac{1}{s^2}\frac{ds}{dt} \quad \rightarrow \quad \boxed{\frac{d}{dt}\left(\frac{1}{s}\right) = -\frac{\dot{s}}{s^2}} \tag{7.2}$$

7.1.2　Vector Derivative Identities

Consider an arbitrary physical vector, s, that is a function of time, t. Below we derive two identities involving time variables of functions of s that will be useful in later derivations. First, recall[1] that

$$s \cdot s = s^2$$

where $s = |s|$ is the magnitude of s. Differentiating both sides with respect to time,

$$\frac{d}{dt}(s \cdot s) = \frac{d}{dt}(s^2) \quad \rightarrow \quad \frac{ds}{dt} \cdot s + s \cdot \frac{ds}{dt} = 2s\frac{ds}{dt}$$

Since the dot product is commutative ($s \cdot y = y \cdot s$) [23, p. 74], [13, pp. 15, 19],

$$2s \cdot \frac{ds}{dt} = 2s\frac{ds}{dt} \quad \rightarrow \quad \boxed{s \cdot \dot{s} = s\dot{s}}$$

Combining these with Eq. (**??**), we get [13, pp. 15]

$$\boxed{s \cdot \dot{s} = s\dot{s} = \frac{d}{dt}\left(\frac{s^2}{2}\right)} \tag{7.3}$$

The derivative property above is scalar equation involving dot products of vectors. Now, we derive another derivative property that is a vector equation involving cross products of vectors.

$$\frac{d}{dt}(s \times \dot{s}) = \left(\frac{ds}{dt} \times \dot{s}\right) + \left(s \times \frac{d\dot{s}}{dt}\right)$$
$$= (\dot{s} \times \dot{s}) + (s \times \ddot{s})$$

Since $s \times s = \mathbf{0}$ for any vector (see Section 2.7), it follows that $\dot{s} \times \dot{s} = \mathbf{0}$. Applying this relationship to this case [13, p. 16], [103, p. 24], [23, p. 72],

$$\frac{d}{dt}(s \times \dot{s}) = \mathbf{0} + (s \times \ddot{s})$$

$$\boxed{\frac{d}{dt}(s \times \dot{s}) = s \times \ddot{s}} \tag{7.4}$$

Finally, we sometimes encounter a derivative of a unit vector, $\hat{s} = s/s$. We can evaluate this derivative directly using the quotient rule.

$$\boxed{\frac{d}{dt}\left(\frac{s}{s}\right) = \frac{s\dot{s} - s\dot{s}}{s^2}} \tag{7.5}$$

7.1.3　Trigonometric Tools

TODO: angle between two vectors: arccos2 TODO: Angle domains and wrapping

7.2　Constants of Integration

TODO: briefly discussed on p. 77 of curtis, it refers to section 2.3

---

[1]　See Eq. (2.25) in Section 2.7

7.3   Conservation of (Specific) Mechanical Energy

Recall the Fundamental Orbital Differential Equation (FODE) given by Eq. (**??**) in Section **??**:

$$\ddot{\boldsymbol{r}} + \mu\frac{\boldsymbol{r}}{r^3} = \boldsymbol{0} \tag{??}$$

Dot-multiplying both sides of the above equation with the inertial velocity, $\boldsymbol{v}$,

$$\boldsymbol{v} \cdot \left(\ddot{\boldsymbol{r}} + \mu\frac{\boldsymbol{r}}{r^3}\right) = \boldsymbol{v} \cdot \boldsymbol{0}$$

Distributing the dot product on the left hand side, and noting that the right hand side evaluates to a scalar 0, we get

$$\boldsymbol{v} \cdot \ddot{\boldsymbol{r}} + \frac{\mu}{r^3}\left(\boldsymbol{v} \cdot \boldsymbol{r}\right) = 0$$

Substituting $\ddot{\boldsymbol{r}} = \dot{\boldsymbol{v}}$ and $\boldsymbol{v} = \dot{\boldsymbol{r}}$ (since velocity is the derivative of position),

$$\boldsymbol{v} \cdot \dot{\boldsymbol{v}} + \frac{\mu}{r^3}\left(\dot{\boldsymbol{r}} \cdot \boldsymbol{r}\right) = 0 \tag{7.6}$$

Equation (7.6) above is already a scalar equation, but its left hand side is written in terms of vectors. Recall from Eq. (7.3) in Section 7.1.2 that for an arbitrary physical vector $\boldsymbol{s}$, the following relationship holds:

$$\boldsymbol{s} \cdot \dot{\boldsymbol{s}} = s\dot{s}$$

Applying this identity to Eq. (7.6), we get

$$v\dot{v} + \frac{\mu}{r^3}\left(r\dot{r}\right) = 0 \quad \rightarrow \quad v\dot{v} + \frac{\mu\dot{r}}{r^2} = 0 \tag{7.7}$$

Using Eq. (7.1), we can write the product $v\dot{v}$ as

$$v\dot{v} = \frac{d}{dt}\left(\frac{v^2}{2}\right) \tag{7.8}$$

Using Eq. (7.2), we can write the product $\mu\dot{r}/r^2$ as

$$\frac{\mu\dot{r}}{r^2} = \mu\left(\frac{\dot{r}}{r^2}\right) = -\mu\left(-\frac{\dot{r}}{r^2}\right) = -\mu\left[\frac{d}{dt}\left(\frac{1}{r}\right)\right] \quad \rightarrow \quad \frac{\mu\dot{r}}{r^2} = \frac{d}{dt}\left(-\frac{\mu}{r}\right) \tag{7.9}$$

Substituting Eqs. (7.8) and (7.9) into Eq. (7.7),

$$\frac{d}{dt}\left(\frac{v^2}{2}\right) + \frac{d}{dt}\left(-\frac{\mu}{r}\right) = 0 \quad \rightarrow \quad \frac{d}{dt}\left(\frac{v^2}{2} - \frac{\mu}{r}\right) = 0 \quad \rightarrow \quad \frac{v^2}{2} - \frac{\mu}{r} = \text{constant} \tag{7.10}$$

Since the right hand side is equal to 0, this equation implies that the argument of the derivative is a constant.

$$\underbrace{\frac{v^2}{2} - \frac{\mu}{r}}_{\mathcal{E}} = \text{constant}$$

We know from Eq. (**??**) that the left hand side is just the specific mechanical energy, $\mathcal{E}$. Thus, we have shown that for motion defined by the FODE, specific mechanical energy is conserved [13, p. 15], [103, p. 27], [23, p. 80].

$$\boxed{\mathcal{E} = \text{constant} \quad \rightarrow \quad \dot{\mathcal{E}} = 0} \tag{7.11}$$

7.3.1   The Energy Integral

In Section **??**, we derived the following equation:

$$\frac{d}{dt}\left(\frac{v^2}{2} - \frac{\mu}{r}\right) = 0$$

We recognized that the left hand side of this equation was equal to the specific mechanical energy ($\mathcal{E}$) as defined in Section **??**, and we determined that it must be a constant since the right hand side of this equation was just 0.

     Alternatively, let's integrate both sides of this equation over time.

$$\int \left[\frac{d}{dt}\left(\frac{v^2}{2} - \frac{\mu}{r}\right)\right] dt = \int 0\, dt \quad \rightarrow \quad \frac{v^2}{2} - \frac{\mu}{r} + C_1 = C_2$$

Choosing $C_1 = 0$ is *identical* to choosing our reference datum for the gravitational potential energy in Section **??**. This results in

$$\frac{v^2}{2} - \frac{\mu}{r} = C_2$$

By defining $\mathcal{E} = C_2$, we simply get

$$\mathcal{E} = \frac{v^2}{2} - \frac{\mu}{r}$$

which is just our definition of the specific mechanical energy from before. However, we now also refer to $\mathcal{E}$ as the **energy integral** since it is a direct result of integrating a manipulated form of the FODE over time [103, p. 27].

7.4   Conservation of (Specific) Angular Momentum

Once again, recall the Fundamental Orbital Differential Equation (FODE) given by Eq. (**??**) in Section **??**:

$$\ddot{\boldsymbol{r}} + \mu\frac{\boldsymbol{r}}{r^3} = \boldsymbol{0} \tag{??}$$

In Section **??**, we dot-multiplied both sides of the FODE by the inertial velocity, $\boldsymbol{v}$. Here, we instead cross-multiply both sides of the FODE by the position, $\boldsymbol{r}$.

$$\boldsymbol{r} \times \left(\ddot{\boldsymbol{r}} + \mu\frac{\boldsymbol{r}}{r^3}\right) = \boldsymbol{r} \times \boldsymbol{0}$$

Distributing the cross product on the left hand side, and noting that $\boldsymbol{r} \times \boldsymbol{0} = \boldsymbol{0}$ on the right hand side, we get

$$\boldsymbol{r} \times \ddot{\boldsymbol{r}} + \frac{\mu}{r^3}(\boldsymbol{r} \times \boldsymbol{r}) = \boldsymbol{0}$$

For an arbitrary vector, $\boldsymbol{s}$, we know that $pvecs \times \boldsymbol{s} = \boldsymbol{0}$ (see Section 2.7). Therefore, the second term vanishes and we are left with

$$\boldsymbol{r} \times \ddot{\boldsymbol{r}} = \boldsymbol{0} \tag{7.12}$$

From Eq. (7.4), we can rewrite the left hand side as a derivative.

$$\frac{d}{dt}(\boldsymbol{r} \times \dot{\boldsymbol{r}}) = \boldsymbol{0}$$

Since $\boldsymbol{v} = \dot{\boldsymbol{r}}$,

$$\frac{d}{dt}(\boldsymbol{r} \times \boldsymbol{v}) = \boldsymbol{0}$$

Since the right hand side is equal to 0, this equation implies that the argument of the derivative is a constant.

$$\underbrace{\boldsymbol{r} \times \boldsymbol{v}}_{\boldsymbol{h}} = \textbf{constant}$$

We know from Eq. (**??**) that the left hand side is just the specific angular momentum, $\boldsymbol{h}$. Thus, we have shown that for motion defined by the FODE, specific angular momentum is conserved [13, pp. 16–17], [103, pp. 23–24], [23, p. 72].

$$\boxed{\boldsymbol{h} = \textbf{constant} \quad \rightarrow \quad \dot{\boldsymbol{h}} = \boldsymbol{0}} \tag{7.13}$$

This also implies that the *magnitude* of the specific angular momentum, $h = |\boldsymbol{h}|$, is conserved.

$$\boxed{h = \text{constant} \quad \rightarrow \quad \dot{h} = 0} \tag{7.14}$$

7.4.1   Integration Constants

TODO

7.5   Solution of the Fundamental Orbital Differential Equation

In this section, we develop (using three different approaches) a *scalar* solution to the Fundamental Orbital Differential Equation (FODE) given by Eq. (**??**) from Section **??**:

$$\ddot{\boldsymbol{r}} + \mu \frac{\boldsymbol{r}}{r^3} = \boldsymbol{0} \tag{??}$$

Let's begin by crossing both sides of the FODE into $\boldsymbol{h}$.

$$\left(\ddot{\boldsymbol{r}} + \mu \frac{\boldsymbol{r}}{r^3} \right) \times \boldsymbol{h} = \boldsymbol{0} \times \boldsymbol{h}$$

Distributing the cross product on the left hand side, and noting that the right hand side evaluates to $\boldsymbol{0}$,

$$(\ddot{\boldsymbol{r}} \times \boldsymbol{h}) + \left(\mu \frac{\boldsymbol{r}}{r^3} \times \boldsymbol{h} \right) = \boldsymbol{0} \tag{7.15}$$

We can notice that the first term, $\ddot{\boldsymbol{r}} \times \boldsymbol{h}$, would appear when using the product rule to differentiate $\dot{\boldsymbol{r}} \times \boldsymbol{h}$ with respect to time. Evaluating this derivative,

$$\frac{d}{dt} \left(\dot{\boldsymbol{r}} \times \boldsymbol{h} \right) = \left(\frac{d\dot{\boldsymbol{r}}}{dt} \times \boldsymbol{h} \right) + \left(\dot{\boldsymbol{r}} \times \frac{d\boldsymbol{h}}{dt} \right)$$

$$= (\ddot{\boldsymbol{r}} \times \boldsymbol{h}) + \left(\dot{\boldsymbol{r}} \times \dot{\boldsymbol{h}} \right)$$

From conservation of specific angular momentum (Eq. (7.13) in Section **??**), we know $\dot{\boldsymbol{h}} = \boldsymbol{0}$. Therefore, the equation above gives us

$$\ddot{\boldsymbol{r}} \times \boldsymbol{h} = \frac{d}{dt} \left(\dot{\boldsymbol{r}} \times \boldsymbol{h} \right)$$

which we can substitute into Eq. (7.15) to get

$$\frac{d}{dt} \left(\dot{\boldsymbol{r}} \times \boldsymbol{h} \right) + \left(\mu \frac{\boldsymbol{r}}{r^3} \times \boldsymbol{h} \right) = \boldsymbol{0} \tag{7.16}$$

Now, the first term in this equation is a time derivative, and the right hand is equal to 0. If we can replace the second term on the left hand side with a time derivative, then we can combine the two derivatives on the left hand side, and their argument will be a constant since the right hand side is $\boldsymbol{0}$. The easiest substitution to identify is that $\boldsymbol{h} = \boldsymbol{r} \times \boldsymbol{v}$. Substituting this into the second term on the left hand side,

$$\mu \frac{\boldsymbol{r}}{r^3} \times \boldsymbol{h} = \mu \frac{\boldsymbol{r}}{r^3} \times (\boldsymbol{r} \times \boldsymbol{v})$$

$$= \frac{\mu}{r^3} [\boldsymbol{r} \times (\boldsymbol{r} \times \boldsymbol{v})]$$

Using the triple product expansion (see Eq. (2.30) in Section 2.7), we can rewrite the vector triple product in terms of dot products.

$$\mu \frac{\boldsymbol{r}}{r^3} \times \boldsymbol{h} = \frac{\mu}{r^3} \left[(\boldsymbol{r} \cdot \boldsymbol{v})\boldsymbol{r} - (\boldsymbol{r} \cdot \boldsymbol{r})\boldsymbol{v} \right]$$

Since velocity is the derivative of position, we can substitute $\boldsymbol{v} = \dot{\boldsymbol{r}}$.

$$\mu \frac{\boldsymbol{r}}{r^3} \times \boldsymbol{h} = \frac{\mu}{r^3} \left[(\boldsymbol{r} \cdot \dot{\boldsymbol{r}})\boldsymbol{r} - (\boldsymbol{r} \cdot \boldsymbol{r})\dot{\boldsymbol{r}} \right]$$

From Eq. (7.3) in Section 7.1.2, we know that $\boldsymbol{r} \cdot \dot{\boldsymbol{r}} = r\dot{r}$. Additionally, we know that $\boldsymbol{r} \cdot \boldsymbol{r} = r^2$. Making these substitutions,

$$\mu \frac{\boldsymbol{r}}{r^3} \times \boldsymbol{h} = \frac{\mu}{r^3} \left(r\dot{r}\boldsymbol{r} - r^2\dot{\boldsymbol{r}} \right)$$
$$= \mu \left(\frac{\dot{r}\boldsymbol{r} - r\dot{\boldsymbol{r}}}{r^2} \right) \tag{7.17}$$

From Eq. (7.5) in Section 7.1.2, we know

$$\frac{d}{dt} \left(\frac{\boldsymbol{r}}{r} \right) = \frac{r\dot{\boldsymbol{r}} - \boldsymbol{r}\dot{r}}{r^2} = - \left(\frac{\boldsymbol{r}\dot{r} - r\dot{\boldsymbol{r}}}{r^2} \right) = - \left(\frac{\dot{r}\boldsymbol{r} - r\dot{\boldsymbol{r}}}{r^2} \right)$$

Substituting this into Eq. (7.17),

$$\mu \frac{\boldsymbol{r}}{r^3} \times \boldsymbol{h} = -\mu \frac{d}{dt} \left(\frac{\dot{r}\boldsymbol{r} - r\dot{\boldsymbol{r}}}{r^2} \right)$$
$$= \frac{d}{dt} \left(-\frac{\mu \boldsymbol{r}}{r} \right) \tag{7.18}$$

Now, we also have the second term in Eq. (7.16) written as a time derivative, as given by Eq. (7.18). Substituting Eq. (7.18) into Eq. (7.16),

$$\frac{d}{dt} (\dot{\boldsymbol{r}} \times \boldsymbol{h}) + \frac{d}{dt} \left(-\frac{\mu \boldsymbol{r}}{r} \right) = \boldsymbol{0}$$

Due to the linearity of the derivative operator, the derivative arguments can be combined to yield

$$\frac{d}{dt} \left(\dot{\boldsymbol{r}} \times \boldsymbol{h} - \frac{\mu \boldsymbol{r}}{r} \right) = \boldsymbol{0}$$

Integrating both sides over time, we get

$$\dot{\boldsymbol{r}} \times \boldsymbol{h} - \frac{\mu \boldsymbol{r}}{r} = \boldsymbol{B} \tag{7.19}$$

where $\boldsymbol{B}$ is a constant vector known as the **Laplace-Runge-Lenz (LRL) vector** [48], [23, p. 75]. Since $\boldsymbol{B}$ is a constant,

$$\boxed{\dot{\boldsymbol{B}} = \boldsymbol{0}} \tag{7.20}$$

At this point, we have a vector solution in terms of LRL vector $\boldsymbol{B}$, which has some unknown value. To obtain a scalar equation, we dot-multiply both side of Eq. (7.19) by the position, $\boldsymbol{r}$.

$$\boldsymbol{r} \cdot \left(\dot{\boldsymbol{r}} \times \boldsymbol{h} - \frac{\mu \boldsymbol{r}}{r} \right) = \boldsymbol{r} \cdot \boldsymbol{B}$$

Distributing the dot product on the left hand side,

$$\boldsymbol{r} \cdot (\dot{\boldsymbol{r}} \times \boldsymbol{h}) - \boldsymbol{r} \cdot \left(\frac{\mu \boldsymbol{r}}{r} \right) = \boldsymbol{r} \cdot \boldsymbol{B} \tag{7.21}$$

Using Eq. (2.28) to reorder the vector operations in the first term, noting that $\dot{\boldsymbol{r}} = \boldsymbol{v}$, and then recognizing the specific angular momentum vector $\boldsymbol{h} = \boldsymbol{r} \times \boldsymbol{h}$,

$$\boldsymbol{r} \cdot (\dot{\boldsymbol{r}} \times \boldsymbol{h}) = (\boldsymbol{r} \times \dot{\boldsymbol{r}}) \cdot \boldsymbol{h} = (\boldsymbol{r} \times \boldsymbol{v}) \cdot \boldsymbol{h} = \boldsymbol{h} \cdot \boldsymbol{h} = h^2 \tag{7.22}$$

Simplifying the second term in Eq. (7.21),

$$\boldsymbol{r} \cdot \left(\frac{\mu \boldsymbol{r}}{r} \right) = \frac{\mu (\boldsymbol{r} \cdot \boldsymbol{r})}{r} = \frac{\mu r^2}{r} = \mu r \tag{7.23}$$

Last but not least, we can write the right hand side of Eq. (7.21) in a scalar form by defining the angle between $\boldsymbol{r}$ and $\boldsymbol{B}$ as ν. This angle, ν, is referred to as the **true anomaly** and will be covered in greater depth later. From the definition of the dot product in Eq. (2.24),

$$\boldsymbol{r} \cdot \boldsymbol{B} = rB \cos \nu \tag{7.24}$$

Substituting Eqs. (7.22), (7.23), and Eq. (7.24) into Eq. (7.21),

$$h^2 - \mu r = rB \cos \nu$$

At this point, we are very close to the solution. Since our primary goal is to solve the FODE for position, we solve the equation above for the position *magnitude*, r. Note that for later convenience, we first divide both sides by μ.

$$\frac{h^2}{\mu} - r = \frac{rB \cos \nu}{\mu} \quad \rightarrow \quad r + \frac{rB \cos \nu}{\mu} = \frac{h^2}{\mu} \quad \rightarrow \quad r\left(1 + \frac{B \cos \nu}{\mu}\right) = \frac{h^2}{\mu}$$

$$r = \frac{h^2/\mu}{1 + [(B \cos \nu)/\mu]} \tag{7.25}$$

Before we wrap up, let's note that everything is a constant except for ν:

1. h is a constant due to conservation of specific angular momentum (see Section 7.4).
2. μ is a constant since we assume the mass of the central body is constant.
3. B is a constant since $\dot{\boldsymbol{B}} = \boldsymbol{0}$ (Eq. (7.20)).

Therefore, r is only a function of ν, and we rewrite Eq. (7.25) in a "cleaner" form as

$$\boxed{r = \frac{h^2/\mu}{1 + (B/\mu) \cos \nu}} \tag{7.26}$$

> Our original goal was to solve the FODE for the position, $\boldsymbol{r}$, as a function of time. However, at this point, we have a *scalar* equation for the position *magnitude*, r, as a function of the true anomaly (ν) and not time. However, ν turns out to be a function of time, which we will cover later in this chapter, and the orbital motion ends up being on a single plane (see Section **??**), which we then use to define the orbit in three dimensions in Chapter **??**.

7.6 The Trajectory Equation: Orbits as Conic Sections

TODO: refer to this as a Keplerian or unperturbed orbit

Recall the solution to the FODE given by (7.26).

$$r = \frac{h^2/\mu}{1 + (B/\mu) \cos \nu} \tag{7.26}$$

Let's define the **eccentricity**, e, and **semiparameter** (also known as the **semi-latus rectum**), p, as

$$\boxed{e = \frac{B}{\mu}} \tag{7.27}$$

$$\boxed{p = \frac{h^2}{\mu}} \tag{7.28}$$

Using these definitions, we rewrite Eq. (7.26) as

$$\boxed{r = \frac{p}{1 + e \cos \nu}} \tag{7.29}$$

Equation (7.29) is known as the **trajectory equation**. We can note that Eq. (7.29) is just a conic section written in polar form. For a conic section in polar form, the radial distance, r, is measured from the prime focus of the conic section. Thus, since the trajectory equation is a conic section where the radial distance is measured from the central body's center, the central body must always be at the prime focus of the orbit [13, p. 23].

> The trajectory equation defines a satellite's orbit around a central body in inertial space as a conic section (in polar form) with its primary focus at the center of mass of the central body.

All conic sections have the following parameters:

Table 7.1: Conic section parameters.

Parameter	Symbol	Description
center	-	center of the conic section [31]
primary focus	F	first point used to construct a conic section [13, p. 22–23]
secondary/vacant focus	F'	second point used to construct a conic section [13, p. 22–23]
semi-major axis	a	distance from the center to a vertex [31]
semi-minor axis	b	distance from the center to a covertex [31]
focal distance[2]	c	distance from the center to a focus [31]
eccentricity	e	constant ratio between the focal distance and the semi-major axis [13, pp. 22–24]
semiparameter[3]	p	distance from the primary focus to the conic section, measured perpendicular to the major axis [103, p. 17]

The first three parameters in Table 7.1 are simply point used to define the conic section. In general, the two foci are placed on either side of the center. The conic section is then formed by moving a point so that its absolute distance from a focus to the directrix. We will not discuss the directrix any further since it is irrelevant to orbital mechanics [103, p. 23], but it is useful to know that the directrix, together with the foci and the eccentricity, are used to define a conic section. The remaining parameters in Table 7.1 are all geometric lengths. A conic section can be fully defined with just two of these parameters (given that we know all the various relationships between the parameters).

In the context of orbital mechanics, we are most interested in the following two conic parameters:

1. Semi-major axis (a): defines the size (i.e. energy) of an orbit (see Section **??**)
2. Eccentricity (e): defines the shape of an orbit

Both of these parameters are general to any conic section, not just in the context of orbital mechanics. While the semi-major axis, a, becomes important for defining the energy of an orbit (see Section **??**), the eccentricity, e, has a greater impact on how we analyze the orbit. This is because for different values of e, we have different types of conic sections, as summarized in Table 7.2 below.

---

[2]   Also referred to as the focal eccentricity [31].
[3]   Also referred to as the semi-latus rectum [103, p. 17].

Table 7.2: Conic section types.

Eccentricity	Conic Section Type	See Section
$e = 0$	circle	7.9
$0 < e < 1$	ellipse	??
$e = 1$	parabola	7.10
$e > 1$	hyperbola	??

7.6.1 Periapsis and Apoapsis

> Many of the equations developed in this section are not valid for parabolic orbits. The special case of the parabolic orbit will be covered in depth in Section 7.10.

The **periapsis** of an orbit is its closest point to the central body, which is located at the primary focus of the conic section representing the orbital trajectory. Conversely, the **apoapsis** of an orbit is its farthest point from the central body. Collectively, these two points are known as the **apsides** of an orbit. The line connecting the two apsides is known as the **line of apsides** and defines the semi-major axis of the conic section.

To find the periapsis and apoapsis locations, let's begin with the trajectory equation given by Eq. (7.29).

$$r = \frac{p}{1 + e \cos \nu} \tag{7.29}$$

Recall that r measures the distance from the primary focus (i.e. the central body) to the satellite position. To find the minimum and maximum values of r, we can follow the standard optimization procedure. Since r is only a function of ν, we differentiate r with respect to ν, and solve for those values of ν that make the derivative equal to 0.

$$\frac{dr}{d\nu} = \frac{(1 + e \cos \nu)\frac{dp}{d\nu} - p\frac{d}{d\nu}(1 + e \cos \nu)}{(1 + e \cos \nu)^2} = \frac{0 - p(-e \sin \nu)}{(1 + e \cos \nu)^2}$$

$$\boxed{\frac{dr}{d\nu} = \frac{pe \sin \nu}{(1 + e \cos \nu)^2}} \tag{7.30}$$

We box Eq. (7.30) since it will be useful for many other derivations [75, p. 18]. Setting $dr/d\nu$ equal to 0,

$$\frac{dr}{d\nu} = \frac{pe \sin \nu}{(1 + e \cos \nu)^2} = 0$$

The derivative is equal to 0 when its numerator is equal to 0.

$$pe \sin \nu = 0 \quad \rightarrow \quad \sin \nu = 0 \tag{7.31}$$

There are two values of ν that satisfy this equation:

$$\nu_{1,2} = 0, \pi$$

Now, we need to determine which value of ν corresponds to the smallest value of r, and which corresponds to the largest value. Substituting both values in to the trajectory equation,

$$r_1 = \frac{p}{1 + e \cos 0} = \frac{p}{1 + e}$$

$$r_2 = \frac{p}{1 + e \cos \pi} = \frac{p}{1 - e}$$

We know from Eq. (7.28) that $e = B/\mu$ must be greater than or equal to 0 since B the magnitude of a vector and μ is a positive constant. For $e \geq 0$, we have that $r_2 \geq r_1$. Thus, we have found the **periapsis radius**, r_p, and the **apoapsis radius**, r_a.

$$\boxed{r_p = \frac{p}{1 + e}} \qquad (\text{at } \nu = 0) \tag{7.32}$$

$$r_a = \frac{p}{1 - e} \qquad \text{(at } \nu = \pi) \tag{7.33}$$

Additionally, we now know where periapsis and apoapsis occur along the orbit.

- periapsis occurs at $\nu = 0$
- apoapsis occurs at $\nu = \pi$

From the geometry of conic sections, we know that by definition, the major axis is the the distance between the closest and furthest points in the trajectory. Additionally, the major axis has twice the length of the semi-major axis, which is defined as a. Finally, from Eqs. (7.32) and (7.33), we have the distances from the primary focus to the closest and furthest points on the orbit. Therefore,

$$2a = r_p + r_a \tag{7.34}$$

As an aside, we can solve for a as

$$a = \frac{r_p + r_a}{2} \tag{7.35}$$

Now, going back and substituting Eqs. (7.32) and (7.33) into Eq. (7.34),

$$2a = \frac{p}{1 + e} + \frac{p}{1 - e}$$

We typically describe orbits using a and e. Therefore, it is useful to develop an expression for p in terms of a and e. Solving for p, [13, pp. 24–25],

$$2a = \frac{p(1 - e)}{(1 + e)(1 - e)} + \frac{p(1 + e)}{(1 + e)(1 - e)} = \frac{p(1 - e) + p(1 + e)}{(1 + e)(1 - e)} = \frac{p - e + p + e}{1 - e + e - e^2} = \frac{2p}{1 - e^2}$$

$$p = a(1 - e^2) \tag{7.36}$$

We can also rearrange this equation in two separate ways to obtain additional forms of this relationship [13, p. 28].

$$a = \frac{p}{1 - e^2} \tag{7.37}$$

$$e = \sqrt{1 - \frac{p}{a}} \tag{7.38}$$

Now that we have an equation for p in terms of a and e, we can also define r_p and r_a in terms of a and e alone. Substituting Eq. (7.36) into Eqs. (7.32) and (7.33),

$$r_p = \frac{a(1 - e^2)}{1 + e} = \frac{a(1 - e)(1 + e)}{1 + e}$$

$$r_p = a(1 - e) \tag{7.39}$$

$$r_a = \frac{a(1 - e^2)}{1 - e} = \frac{a(1 - e)(1 + e)}{1 - e} =$$

$$r_a = a(1 + e) \tag{7.40}$$

We can also get two convenient expressions for a from Eqs. (7.39) and (7.40).

$$a = \frac{r_p}{1 - e} \tag{7.41}$$

$$a = \frac{r_a}{1 + e} \tag{7.42}$$

Equation Eqs. (7.41) and (7.42), we can solve for e directly in terms of r_p and r_a [13, pp. 24–25], [75, pp. 15–16], [13, pp. 30–31].

$$\frac{r_p}{1-e} = \frac{r_a}{1+e} \quad \to \quad r_a(1-e) = r_p(1+e) \quad \to \quad r_a - r_a e = r_p + r_p e$$

$$r_a - r_p = r_a e + r_p e \quad \to \quad r_a - r_p = (r_a + r_p)e$$

$$\boxed{e = \frac{r_a - r_p}{r_a + r_p}} \tag{7.43}$$

Radial rates at the apsides

The **radial rate**, $\dot{r}$, at any point in the orbit is simply

$$\dot{r} = \frac{dr}{dt}$$

However, the trajectory equation given by Eq. (7.29) defines r as a function of the true anomaly, ν. Therefore, we can use the chain rule to obtain

$$\dot{r} = \frac{dr}{dt} = \frac{dr}{d\nu}\frac{d\nu}{dt}$$

Substituting Eq. (7.30),

$$\dot{r} = \left[\frac{pe\sin\nu}{(1+e\cos\nu)^2}\right]\frac{d\nu}{dt} \tag{7.44}$$

Despite not having any knowledge of $d\nu/dt$, we can still solve for $\dot{r}$ at periapsis ($\nu = 0$) and apoapsis ($\nu = \pi$). let $\dot{r}_p$ be the **periapsis radial rate** and $\dot{r}_a$ be the **apoapsis radial rate**. Then

$$\dot{r}_p = \dot{r}(0) = \left[\frac{pe\sin 0}{(1+e\cos 0)^2}\right]\frac{d\nu}{dt} = 0$$

$$\dot{r}_a = \dot{r}(\pi) = \left[\frac{pe\sin\pi}{(1+e\cos 0)^2}\right]\frac{d\nu}{dt} = 0$$

since $\sin 0 = \sin\pi = 0$. Thus, we have that the radial rates at both of the apsides are 0.

$$\boxed{\dot{r}_p = \dot{r}_a = 0} \tag{7.45}$$

> We will discuss radial rates again in Section 7.6.5, and then cover radial rates specific to elliptical and circular orbits in Sections 7.8.1 and TODO, respectively.

Alternative form of the trajectory equation

Recall the trajectory equation defined by Eq. (7.29). We can substitute our new expression for p given by Eq. (7.36) to get an alternative form of the trajectory equation.

$$\boxed{r = \frac{a(1-e^2)}{1+e\cos\nu}} \tag{7.46}$$

7.6.2   Eccentricity Vector

In Eq. (7.27), we defined the eccentricity as the magnitude of the LRL vector, $B = |\boldsymbol{B}|$, divided by the gravitational parameter, μ.

$$e = \frac{B}{\mu}$$

Since μ is a scalar, it follows that we can define a vector quantity

$$\boldsymbol{e} = \frac{\boldsymbol{B}}{\mu} \tag{7.47}$$

where the magnitude of $\boldsymbol{e}$ is the eccentricity, e.

$$\boxed{e = |\boldsymbol{e}|} \tag{7.48}$$

We refer to this vector, $\boldsymbol{e}$, as the **eccentricity vector**.

However, we still don't know what B is, so how can we determine $\boldsymbol{e}$? Recall that in our solution to the FODE, the last vector equation we had was Eq. (7.19). This equation gives us a direct equation for $\boldsymbol{B}$:

$$\boldsymbol{B} = \dot{\boldsymbol{r}} \times \boldsymbol{h} - \frac{\mu \boldsymbol{r}}{r}$$

Since $\boldsymbol{e} = \boldsymbol{B}/\mu$,

$$\boldsymbol{e} = \frac{\boldsymbol{B}}{\mu} = \frac{1}{\mu}\left(\dot{\boldsymbol{r}} \times \boldsymbol{h} - \frac{\mu \boldsymbol{r}}{r}\right)$$
$$= \frac{\dot{\boldsymbol{r}} \times \boldsymbol{h}}{\mu} - \frac{\boldsymbol{r}}{r}$$

Finally, recall that $\dot{\boldsymbol{r}} = \boldsymbol{v}$.

$$\boxed{\boldsymbol{e} = \frac{\boldsymbol{v} \times \boldsymbol{h}}{\mu} - \frac{\boldsymbol{r}}{r}} \tag{7.49}$$

We can also write $\boldsymbol{e}$ in terms of $\boldsymbol{r}$, $\boldsymbol{v}$, and μ alone by substituting $\boldsymbol{h} = \boldsymbol{r} \times \boldsymbol{v}$.

$$\boldsymbol{e} = \frac{\boldsymbol{v} \times (\boldsymbol{r} \times \boldsymbol{v})}{\mu} - \frac{\boldsymbol{r}}{r}$$

Applying the triple product expansion from Eq. (2.30) in Section 2.7,

$$\boldsymbol{e} = \frac{(\boldsymbol{v} \cdot \boldsymbol{v})\boldsymbol{r} - (\boldsymbol{r} \cdot \boldsymbol{v})\boldsymbol{v}}{\mu} - \frac{\boldsymbol{r}}{r}$$

Since $\boldsymbol{v} \cdot \boldsymbol{v} = v^2$ [13, pp. 25–26],

$$\boldsymbol{e} = \frac{v^2 \boldsymbol{r} - (\boldsymbol{r} \cdot \boldsymbol{v})\boldsymbol{v}}{\mu} - \frac{\boldsymbol{r}}{r}$$

$$\boxed{\boldsymbol{e} = \left(\frac{v^2}{\mu} - \frac{1}{r}\right)\boldsymbol{r} - \left(\frac{\boldsymbol{r} \cdot \boldsymbol{v}}{\mu}\right)\boldsymbol{v}} \tag{7.50}$$

Direction of the Eccentricity Vector

Recall from Section 7.5 that the true anomaly, ν, is defined as the angle between the LRL vector, $\boldsymbol{B}$, and the position vector, $\boldsymbol{r}$. This implies that when $\nu = 0$, $\boldsymbol{B}$ and $\boldsymbol{r}$ are collinear. We also know that the satellite is at periapsis when $\nu = 0$ (see our discussion in the previous section, Section 7.6.1). Thus,

> The eccentricity vector points towards periapsis.

Units

Just by looking at the second term of (7.49), which is the unit vector of position, we can see that the eccentricity vector is unitless (since unit vectors are always unitless).

Algorithms

Equations 7.49 and (7.50) give us two equations for obtaining the eccentricity vector from a position and a velocity. We formalize these computations as Algorithms 50 and 49 below. Note that Algorithm 49 makes use of the fact that $s^2 = s \cdot s$ for an arbitrary column vector s.

Algorithm 48: `rvh2e_vec`

Eccentricity vector from position, velocity, and specific angular momentum.

Inputs:
- $[r]_\mathcal{I} \in \mathbb{R}^3$ - position expressed in an inertial frame [m]
- $[v]_\mathcal{I} \in \mathbb{R}^3$ - inertial velocity expressed in an inertial frame [m/s]
- $[h]_\mathcal{I} \in \mathbb{R}^3$ - inertial specific angular momentum expressed in an inertial frame $[\mathrm{m}^2/\mathrm{s}]$
- $\mu \in \mathbb{R}$ - standard gravitational parameter $[\mathrm{m}^3/\mathrm{s}^2]$

Procedure:

$$[e]_\mathcal{I} = \left(\frac{[v]_\mathcal{I} \times [h]_\mathcal{I}}{\mu} \right) - \left(\frac{[r]_\mathcal{I}}{\|[r]_\mathcal{I}\|} \right)$$

return $[e]_\mathcal{I}$

Outputs:
- $[e]_\mathcal{I} \in \mathbb{R}^3$ - eccentricity vector expressed in the inertial frame

Test Cases:
- See Appendix A.5.1.

Algorithm 49: `rv2e_vec`

Eccentricity vector from position and velocity.

Inputs:
- $[r]_\mathcal{I} \in \mathbb{R}^3$ - position expressed in an inertial frame [m]
- $[v]_\mathcal{I} \in \mathbb{R}^3$ - inertial velocity expressed in an inertial frame [m/s]
- $\mu \in \mathbb{R}$ - standard gravitational parameter $[\mathrm{m}^3/\mathrm{s}^2]$

Procedure:

$$[e]_\mathcal{I} = \left(\frac{[v]_\mathcal{I} \cdot [v]_\mathcal{I}}{\mu} - \frac{1}{\|[r]_\mathcal{I}\|} \right) [r]_\mathcal{I} - \left(\frac{[r]_\mathcal{I} \cdot [v]_\mathcal{I}}{\mu} \right) [v]_\mathcal{I}$$

return $[e]_\mathcal{I}$

Outputs:

- $[e]_\mathcal{I} \in \mathbb{R}^3$ - eccentricity vector expressed in the inertial frame

Test Cases:
- See Appendix A.5.1.

These algorithms give us the eccentricity vector. However, since the eccentricity is just the magnitude of the eccentricity vector, we introduce Algorithms 50 and 51 to calculate the eccentricity.

Algorithm 50: `rvh2e`
Eccentricity from position, velocity, and specific angular momentum.

Inputs:
- $[r]_\mathcal{I} \in \mathbb{R}^3$ - position expressed in an inertial frame $[\text{m}]$
- $[v]_\mathcal{I} \in \mathbb{R}^3$ - inertial velocity expressed in an inertial frame $[\text{m/s}]$
- $[h]_\mathcal{I} \in \mathbb{R}^3$ - inertial specific angular momentum expressed in an inertial frame $[\text{m}^2/\text{s}]$
- $\mu \in \mathbb{R}$ - standard gravitational parameter $[\text{m}^3/\text{s}^2]$

Procedure:
1. Find the eccentricity vector expressed in the inertial frame (Algorithm 48).

$$[e]_\mathcal{I} = \texttt{rvh2e\_vec}\left([r]_\mathcal{I}, [v]_\mathcal{I}, [h]_\mathcal{I}, \mu\right)$$

2. Compute the eccentricity.

$$e = \|[e]_\mathcal{I}\|$$

3. Return the result.

return e

Outputs:
- $e \in \mathbb{R}$ - eccentricity

Test Cases:
- See Appendix A.5.1.

Algorithm 51: `rv2e`
Eccentricity from position and velocity.

Inputs:
- $[r]_\mathcal{I} \in \mathbb{R}^3$ - position expressed in an inertial frame $[\text{m}]$
- $[v]_\mathcal{I} \in \mathbb{R}^3$ - inertial velocity expressed in an inertial frame $[\text{m/s}]$
- $\mu \in \mathbb{R}$ - standard gravitational parameter $[\text{m}^3/\text{s}^2]$

Procedure:

1. Find the eccentricity vector expressed in the inertial frame (Algorithm 49).

$$[e]_\mathcal{I} = \texttt{rvh2e\_vec}\,([r]_\mathcal{I}, [v]_\mathcal{I}, \mu)$$

2. Compute the eccentricity.

$$e = \|[e]_\mathcal{I}\|$$

3. Return the result.

return e

Outputs:

- $e \in \mathbb{R}$ - eccentricity

Test Cases:

- See Appendix A.5.1.

7.6.3 Orbital Plane

Recall from Section 7.4 that the specific angular momentum vector is conserved.

$$h = r \times v = \textbf{constant}$$

This equation immediately implies the following:

1. The direction (and magnitude) of the specific angular momentum, h, is constant.
2. The specific angular momentum (h) of the satellite's orbit is orthogonal (i.e. normal or perpendicular) to both the position (r) and velocity (v) of the satellite.

Since r and v are both always orthogonal to h, and since h is always pointing in the same direction, the satellite's orbit lies on a constant plane that is spanned by the satellite's position and velocity vectors. Since h is orthogonal to r and v, it defines the normal direction of the orbital plane.

The orbital plane is defined entirely by the specific angular momentum vector (h) of the orbit, which serves as its normal vector. The position (r) and velocity (v) of the satellite always lie in the orbital plane.

Furthermore, recall Eq. (**??**) defining the eccentricity vector.

$$e = \left(\frac{v^2}{\mu} - \frac{1}{r}\right) r - \left(\frac{r \cdot v}{\mu}\right) v \tag{??}$$

The eccentricity vector is simply a linear combination of the position and velocity vectors, which both exist entirely in the orbital plane. Thus [23, p. 76],

The eccentricity vector (e) lies in the orbital plane.

Figure 7.1 illustrates the geometry of the orbital plane. On it we can see:

- eccentricity vector (e) pointing towards periapsis (discussed in Section **??**)
- specific angular momentum vector (h) normal to the orbital plane
- true anomaly (ν), as the angle between the eccentricty vector (e) and the position vector (r), measured as a counterclockwise rotation about the axis defined by h
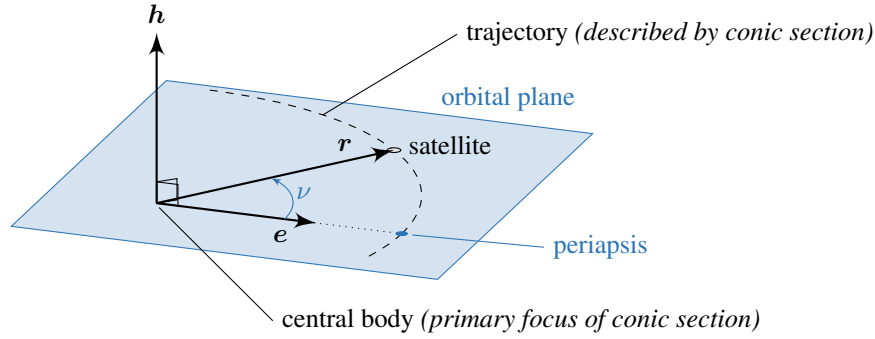
Figure 7.1: Orbital plane.

7.6.4   Perifocal (PQW) Frame

The **perifocal frame** is an inertial coordinate frame with coordinate axes PQW, defined by the ordered basis $(\hat{\boldsymbol{P}}, \hat{\boldsymbol{Q}}, \hat{\boldsymbol{W}})$; hence, it is sometimes referred to as the **PQW frame**. The origin is at the center of mass of the central body, which is also the primary focus of the conic section representing the orbit trajectory. The directions are defined as follow:

1. $\hat{\boldsymbol{P}}$: Points towards periapsis. Since the eccentricity vector, $\boldsymbol{e}$, also points towards periapsis, $\hat{\boldsymbol{P}}$ is just the unit vector in the direction of $\boldsymbol{e}$.
2. $\hat{\boldsymbol{Q}}$: Perpendicular to $\hat{\boldsymbol{P}}$, in the direction of the true anomaly (i.e. in the direction of the satellite motion).
3. $\hat{\boldsymbol{W}}$: Collinear to the specific angular momentum vector (and thus orthogonal to the orbital plane). *Note $\hat{\boldsymbol{W}}$ is defined such that a positive ν corresponds to a positive rotation about the W axis.*

    In practice, we can easily identify $\boldsymbol{e}$ and $\boldsymbol{h}$, so we can just define $\hat{\boldsymbol{P}}$ and $\hat{\boldsymbol{W}}$ in the directions of $\boldsymbol{e}$ and $\boldsymbol{h}$, respectively, and then just let $\hat{\boldsymbol{Q}}$ complete the right-handed triad. The perifocal frame is visualized in Fig. 7.2 below [13, pp. 56–57], [103, pp. 155-156]. As mentioned previously, since $\hat{\boldsymbol{P}}$ points towards periapsis, it is collinear with
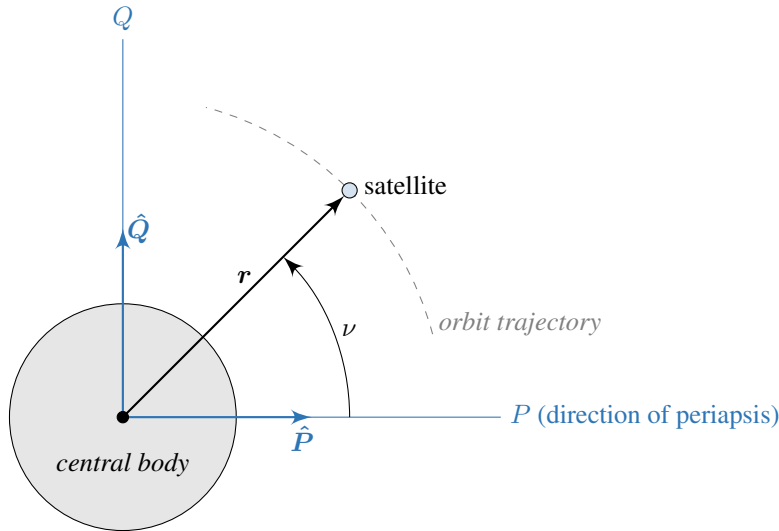
Figure 7.2: Perifocal frame.

the eccentricity vector, $\boldsymbol{e}$, and we can therefore define it as

$$\hat{\boldsymbol{P}} = \frac{\boldsymbol{e}}{e}$$

(7.51)

Conversely, we can write the eccentricity vector in terms of its magnitude, e, and direction, $\hat{\boldsymbol{P}}$.

$$\boxed{\boldsymbol{e} = e\hat{\boldsymbol{P}}}$$ (7.52)

Figure 7.2 presented a two-dimensional view of the perifocal frame. To see its third axis, refer to Fig. 7.3, which presents a more three-dimensional view.

> Since $\hat{\boldsymbol{P}}$ points towards periapsis, and since $\hat{\boldsymbol{W}}$ is collinear with $\boldsymbol{h}$ which is the normal to the orbital plane, the PQ plane defines the orbital plane.
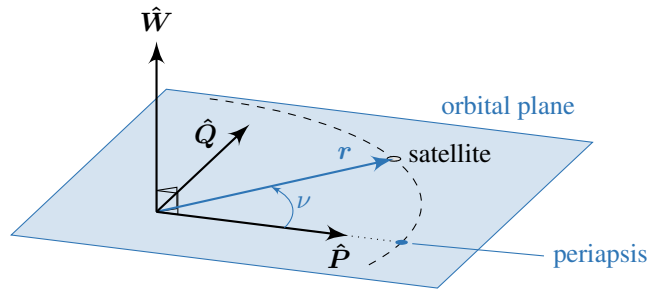
Figure 7.3: Perifocal frame on the orbital plane.

7.6.5   RSW Frame (RTN, LVLH)

Recall Section **??** where we discussed planar motion using polar coordinates. Planar motion in polar coordinates is *exactly* what the trajectory equation defines. In Section **??**, polar coordinates helped define a rotating coordinate frame where the radial direction, $\hat{\boldsymbol{r}}$, was defined by the position vector of a point P, and the transverse direction, $\hat{\boldsymbol{\theta}}$, was normal to the position vector. As a refresher, we include Fig. 7.4 here.
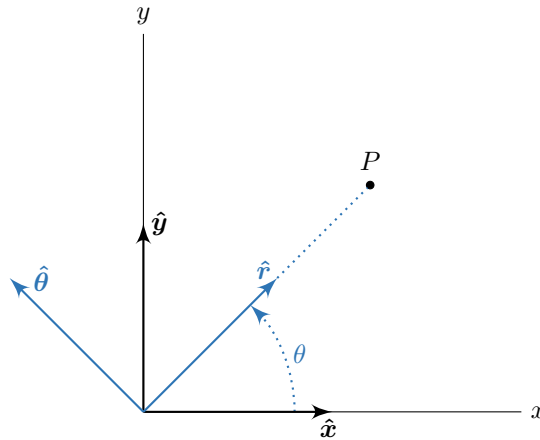
Figure 7.4: Coordinate frame definitions for polar coordinates.

In orbital mechanics, the **RSW frame** takes the place of the rotating frame, and is defined as a rotation of ν about the W axis of the perifocal frame. Additionally, the RSW frame has its origin at the center of mass of the satellite. The RSW frame is defined by the $(\hat{\boldsymbol{R}}, \hat{\boldsymbol{S}}, \hat{\boldsymbol{W}})$ ordered basis [13, pp. 398-399], forming the RSW coordinate system [103, pp. 155-156].
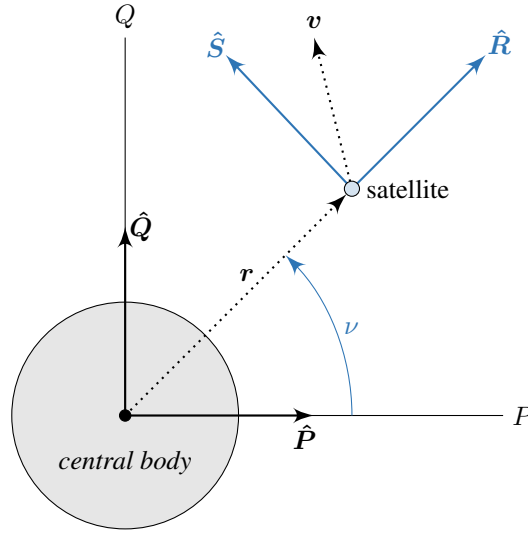
Figure 7.5: RSW and perifocal frames.

1. $\hat{\boldsymbol{R}}$: **Radial direction.** Unit vector collinear to the satellite position vector.
2. $\hat{\boldsymbol{S}}$: **Along-track direction.** Unit vector in the orbital plane normal to $\hat{\boldsymbol{R}}$, in the direction of the true anomaly (i.e. in the direction of the satellite motion).
3. $\hat{\boldsymbol{W}}$: **Cross-track direction.** Unit vector collinear to the specific angular momentum vector (and thus orthogonal to the orbital plane).

The RSW frame also goes by many other names. Some of these include the **satellite frame**, **LVLH frame** (for local-horizontal, local-vertical; see Fig. 7.6 later in this section), **RTN frame** (with radial, transverse, and normal directions) [103, pp. 155-156], and the **Euler-Hill frame** [4, pp. 60–61].

Let's consider the position and velocity of the satellite. Since the radial direction is defined by the position
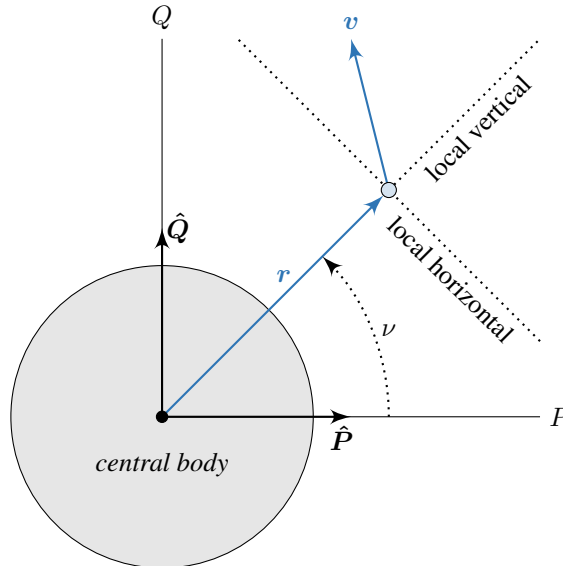
Figure 7.6: Satellite position and velocity.

vector of the satellite, we immediately have

$$\hat{\boldsymbol{R}} = \frac{\boldsymbol{r}}{r} \tag{7.53}$$

Earlier, when listing the three directions of the RSW frame, we listed the $\hat{S}$ direction before the $\hat{W}$ direction. However, in practice, we define the the cross-track direction, $\hat{W}$, first, since we know its the unit vector in the direction of the specific angular momentum vector.

$$\boxed{\hat{W} = \frac{r \times v}{|r \times v|} = \frac{\mathbf{h}}{h} = \hat{h}} \tag{7.54}$$

Note that this is exactly the same $\hat{W}$ as in the perifocal frame (see Section **??**). The along-track direction simply completes the right-handed triad [103, pp. 164].

$$\boxed{\hat{S} = \hat{W} \times \hat{R}} \tag{7.55}$$

> The along-track (or transverse) direction is **not** defined as being tangent to the orbit (the one exception is for circular orbits).

Rotation matrices between perifocal and RSW frames

Since the RSW frame is just a rotation by ν about the 3rd axis (W) of the perifocal frame, the passive rotation matrix from the perifocal frame to the RSW frame, $\mathbf{R}_{\mathrm{pqw} \rightarrow \mathrm{rsw}}$ is simply

$$\boxed{\mathbf{R}_{\mathrm{pqw} \rightarrow \mathrm{rsw}} = \mathbf{R}_3(\nu) = \begin{bmatrix} \cos\nu & \sin\nu & 0 \\ -\sin\nu & \cos\nu & 0 \\ 0 & 0 & 1 \end{bmatrix}} \tag{7.56}$$

It is trivial to find the rotation matrix for the opposite rotation.

$$\boxed{\mathbf{R}_{\mathrm{rsw} \rightarrow \mathrm{pqw}} = \mathbf{R}_{\mathrm{pqw} \rightarrow \mathrm{rsw}}^T = \mathbf{R}_3(-\nu) = \begin{bmatrix} \cos\nu & -\sin\nu & 0 \\ \sin\nu & \cos\nu & 0 \\ 0 & 0 & 1 \end{bmatrix}} \tag{7.57}$$

We formalize Eqs. (7.56) and (7.57) as Algorithms 52 and 53, respectively,

---

Algorithm 52: `rot_pqw2rsw`

Passive rotation matrix from the perifocal (PQW) frame to the RSW frame

Inputs:
- $\nu \in \mathbb{R}$ - true anomaly $[\mathrm{rad}]$

Procedure:

$\mathbf{R}_{\mathrm{pqw} \rightarrow \mathrm{rsw}} = \mathtt{rot3}(\nu)$ (Algorithm 3)

return $\mathbf{R}_{\mathrm{pqw} \rightarrow \mathrm{rsw}}$

Outputs:
- $\mathbf{R}_{\mathrm{pqw} \rightarrow \mathrm{rsw}} \in \mathbb{R}^{3 \times 3}$ - passive rotation matrix from the perifocal (PQW) frame to the RSW frame

Test Cases:
- See Appendix **??**.

Algorithm 53: `rot_rsw2pqw`
Passive rotation matrix from the RSW frame to the perifocal (PQW) frame.

Inputs:
- $\nu \in \mathbb{R}$ - true anomaly $[\mathrm{rad}]$

Procedure:

$$\mathbf{R}_{\mathrm{rsw} \to \mathrm{pqw}} = \texttt{rot3}(-\nu) \qquad \text{(Algorithm 3)}$$
$$\textbf{return } \mathbf{R}_{\mathrm{rsw} \to \mathrm{pqw}}$$

Outputs:
- $\mathbf{R}_{\mathrm{rsw} \to \mathrm{pqw}} \in \mathbb{R}^{3\times 3}$ - passive rotation matrix from the RSW frame to the perifocal (PQW) frame

Test Cases:
- See Appendix **??**.

In Section 7.6.6, we will develop the transformation of kinematics between the RSW and perifocal frames.

Translational kinematics

The translational kinematics we define here are relative to an inertial frame (such as the perifocal or ECI frames). Thus, we continue using the simplified notation defined by Convention 38 in Section 6.1.

Recall the radial and transverse components we introduced in Section **??** when dealing with planar motion using polar coordinates, which the trajectory equation falls squarely under. In Section **??**, we used $\hat{\boldsymbol{R}}$ and $\hat{\boldsymbol{S}}$ to define the radial and transverse directions, respectively. In the context of orbital mechanics, these basis vector correspond to the radial ($\hat{\boldsymbol{R}}$) and along-track ($\hat{\boldsymbol{S}}$) basis vectors of the RSW frame. First, we can rewrite Eqs. (**??**), (**??**), and (**??**) (from Section **??**) for the orbital case, replacing $\theta = \nu$, $\dot{\theta} = \dot{\nu}$, $\ddot{\theta} = \ddot{\nu}$, $\hat{\boldsymbol{r}} = \hat{\boldsymbol{R}}$, and $\hat{\boldsymbol{\theta}} = \hat{\boldsymbol{S}}$. Additionally, we drop the frame levels since all kinematics here are relative to an inertial frame (even if they are expressed in the RSW frame which is noninertial).

$$\boxed{\boldsymbol{r} = r\hat{\boldsymbol{R}}} \tag{7.58}$$

$$\boxed{\boldsymbol{v} = \dot{\boldsymbol{r}} = \dot{r}\hat{\boldsymbol{R}} + r\dot{\nu}\hat{\boldsymbol{S}}} \tag{7.59}$$

$$\boxed{\boldsymbol{a} = \ddot{\boldsymbol{r}} = \left(\ddot{r} - r\dot{\nu}^2\right)\hat{\boldsymbol{R}} + \left(r\ddot{\nu} + 2\dot{r}\dot{\nu}\right)\hat{\boldsymbol{S}}} \tag{7.60}$$

We can also express these equations in the RSW frame to write them in a coordinate vector form.

$$\boxed{[\boldsymbol{r}]_{\mathrm{rsw}} = \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix}} \tag{7.61}$$

$$[\boldsymbol{v}]_{\mathrm{rsw}} = [\dot{\boldsymbol{r}}]_{\mathrm{rsw}} = \begin{bmatrix} \dot{r} \\ r\dot{\nu} \\ 0 \end{bmatrix} \qquad (7.62)$$

$$[\boldsymbol{a}]_{\mathrm{rsw}} = [\ddot{\boldsymbol{r}}]_{\mathrm{rsw}} = \begin{bmatrix} \ddot{r} - r\dot{\nu}^2 \\ r\ddot{\nu} + 2\dot{r}\dot{\nu} \\ 0 \end{bmatrix} \qquad (7.63)$$

Equations (7.61), (7.62), and (7.63) are analogous to Eqs. (**??**), (**??**), and (**??**) (from Section **??**), respectively.

Attitude (rotational) kinematics of the RSW frame

Equations (**??**) and (**??**) from Section **??** defined the angular velocity and acceleration of the rotating frame, $\mathcal{R}$, with respect to the inertial frame, $\mathcal{I}$. In this case, the rotation frame is the RSW frame while the inertial frame can still be a generic inertial frame, $\mathcal{I}$. Additionally, in this case, ν replaces θ.

$$\boxed{{}^{\mathrm{rsw}/\mathcal{I}}\boldsymbol{\omega} = \dot{\nu}\hat{\boldsymbol{W}}} \qquad (7.64)$$

$$\boxed{{}^{\mathrm{rsw}/\mathcal{I}}\boldsymbol{\alpha} = \ddot{\nu}\hat{\boldsymbol{W}}} \qquad (7.65)$$

We can also express these equations in the RSW frame to get

$$\boxed{\left[{}^{\mathrm{rsw}/\mathcal{I}}\boldsymbol{\omega}\right]_{\mathrm{rsw}} = \begin{bmatrix} 0 \\ 0 \\ \dot{\nu} \end{bmatrix}} \qquad (7.66)$$

$$\boxed{\left[{}^{\mathrm{rsw}/\mathcal{I}}\boldsymbol{\alpha}\right]_{\mathrm{rsw}} = \begin{bmatrix} 0 \\ 0 \\ \ddot{\nu} \end{bmatrix}} \qquad (7.67)$$

Equations (7.66) and (7.67) are analogous to Eqs. (**??**) and (**??**) from Section **??**, respectively.

True anomaly rate

The translational kinematics expressed in the RSW frame provide a convenient means for deriving the **true anomaly rate**, which is simply the time derivative of the true anomaly. Recall that the (inertial) specific angular momentum is given by

$$\boldsymbol{h} = \boldsymbol{r} \times \boldsymbol{v}$$

Substituting Eqs. (7.58) and (7.59),

$$\boldsymbol{h} = \left(r\hat{\boldsymbol{R}}\right) \times \left(\dot{r}\hat{\boldsymbol{R}} + r\dot{\nu}\hat{\boldsymbol{S}}\right) = r\dot{r}\left(\hat{\boldsymbol{R}} \times \hat{\boldsymbol{R}}\right) + r^2\dot{\nu}\left(\hat{\boldsymbol{R}} \times \hat{\boldsymbol{S}}\right)$$
$$= r^2\dot{\nu}\hat{\boldsymbol{W}}$$

Taking the magnitude,

$$h = |\boldsymbol{h}| = \left|r^2\dot{\nu}\hat{\boldsymbol{W}}\right|$$

$$\boxed{h = r^2\dot{\nu}} \qquad (7.68)$$

Rearranging to solve for $\dot{\nu}$ [75, p. 18], [103, pp. 24], [13, p. 399], [23, pp. 77],

$$\boxed{\dot{\nu} = \frac{h}{r^2}} \qquad (7.69)$$

Note that h is constant for any Keplerian orbit. Also note that Eq. (**??**) could have been directly found from Eq. (**??**), which can be found in Section **??**.

Radial rate

Now that we know $\dot{\nu}$, we can also find an expression for the radial rate, $\dot{r}$. Eq. (repeated below for convenience) from Section 7.6.1 gives us

$$\dot{r} = - \left[\frac{e \sin \nu}{(1 + e \cos \nu)^2} \right] \frac{d\nu}{dt} \tag{7.6.5}$$

Since $d\nu/dt = \dot{\nu}$, we can just substitute Eq. (7.69).

$$\dot{r} = \left[\frac{e \sin \nu}{(1 + e \cos \nu)^2} \right] \dot{\nu} = \left[\frac{e \sin \nu}{(1 + e \cos \nu)^2} \right] \frac{h}{r^2} \tag{7.70}$$

Squaring the trajectory equation,

$$r = \frac{p}{1 + e \cos \nu} = \quad \rightarrow \quad r^2 = \frac{p^2}{(1 + e \cos \nu)^2}$$

Substituting this into Eq. (7.70) [75, p. 19],

$$\dot{r} = \left[\frac{e \sin \nu}{(1 + e \cos \nu)^2} \right] \dot{\nu} = \left[\frac{e \sin \nu}{(1 + e \cos \nu)^2} \right] \left[\frac{h(1 + e \cos \nu)^2}{p^2} \right] = \frac{he \sin \nu}{p}$$

Substituting Eq. (**??**) for p [23, p. 78],

$$\dot{r} = \frac{he \sin \nu}{h^2/\mu}$$

$$\boxed{\dot{r} = \frac{\mu e \sin \nu}{h}} \tag{7.71}$$

> Radial rates specific to elliptical and circular orbits will be covered in Sections 7.8.1 and TODO, respectively.

Radial and along-track velocity components

The components of the inertial velocity expressed in the RSW frame prove to be extremely useful for deriving fundamental relationships in orbital mechanics. Consider the velocity components shown in Fig. 7.7 The **radial velocity component** (in the $\hat{R}$ direction) and the **along-track (transverse) velocity component** (in the $\hat{S}$ direction) can be extracted directly from either Eq. (7.58) or (7.61) [13, p. 398], [23, p. 77].

$$\boxed{v_R = \dot{r}} \tag{7.72}$$

$$\boxed{v_S = r\dot{\nu}} \tag{7.73}$$

Since the radial velocity component, v_R, is equivalent to the radial rate (see Eq. (7.72)), we can substitute Eq. (**??**) into Eq. (7.72) to get

$$\boxed{v_R = \frac{\mu e \sin \nu}{h}} \tag{7.74}$$

We can also substitute Eq. (7.69) into Eq. (7.73) to find another expression for the along-track velocity component.

$$v_S = r \left(\frac{h}{r^2} \right)$$
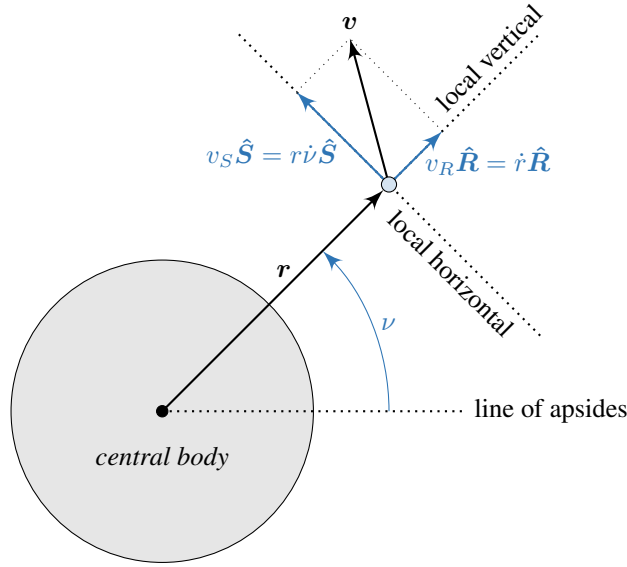
Figure 7.7: Radial and along-track velocity components [103, p. 25], [13, p. 32].

$$\boxed{v_S = \frac{h}{r}} \tag{7.75}$$

Substituting the trajectory equation,

$$v_S = \frac{h(1 + e\cos\nu)}{p}$$

Since $p = h^2/\mu$ (by definition from Eq. (7.28)) [23, p. 77],

$$v_S = \frac{h(1 + e\cos\nu)}{h^2/\mu}$$

$$\boxed{v_S = \frac{\mu(1 + e\cos\nu)}{h}} \tag{7.76}$$

TODO: rewrite fully in terms of orbital elements TODO: position and velocity in perifocal frame: p. 47-48 of rao

7.6.6   Flight Path Angle

TODO: RSW in terms of FPA

The **zenith angle**, β, is the angle between the position and velocity vectors. In orbital/flight mechanics, we typically use the **flight path angle**, which is defined as the complement of the zenith angle[4].

$$\boxed{\gamma = \frac{\pi}{2} - \beta} \tag{7.77}$$

More qualitatively, the flight path angle is the angle between the velocity vector and the local horizontal. The local horizontal is simplify defined normal to the local vertical, which in turn is defined to be in the direction of the position

---

[4]   In many standard texts, such as [103] and [13], the flight path angle is often denoted as ϕ or ϕ_{fpa}, while the zenith angle is denoted γ. Here, we use γ for the flight path angle and β for the zenith angle. γ is often used to represent the flight path angle for aircraft [3], while β has been used by NASA to refer to the zenith angle [19, pp. 3-6 to 3-7]. Note that [19, pp. 3-6 to 3-7] actually refers to the zenith angle as the flight path angle; thus, care must be taken when referring to other resources for the flight path angle [40, 106]. Here, we use the "standard" definition.

vector. A positive flight path angle means that the velocity is pointing "up" above the local horizontal, while a negative flight path angle implies that the velocity is point "down" below the local horizontal, towards the central body [13, pp. 17–18].
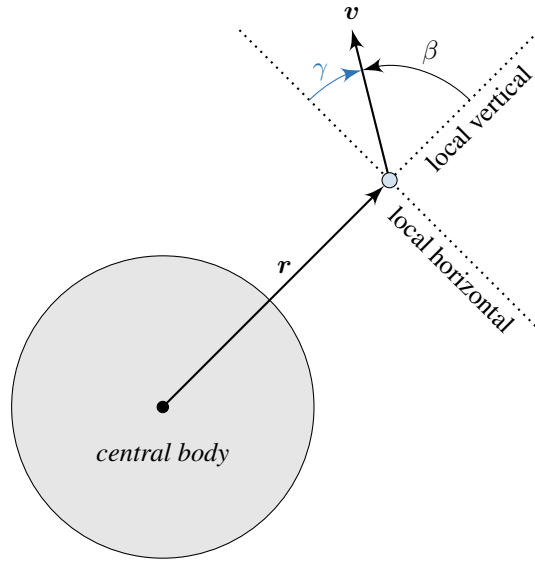
Figure 7.8: Geometry of the flight path angle [13, p. 17]

Flight path angle from position and velocity

Since β is the angle between $\boldsymbol{r}$ and $\boldsymbol{v}$, from the definition of the dot product (see Eq. (2.24) in Section 2.7) we have

$$\boldsymbol{r} \cdot \boldsymbol{v} = rv \cos \beta \tag{7.78}$$

Rearranging Eq. (7.77) to solve for β,

$$\beta = \frac{\pi}{2} - \gamma \tag{7.79}$$

Substituting this into Eq. (7.78),

$$\boldsymbol{r} \cdot \boldsymbol{v} = rv \cos \left(\frac{\pi}{2} - \gamma \right)$$

$$\boldsymbol{r} \cdot \boldsymbol{v} = rv \sin \gamma$$

Solving for γ [75, pp. 20–21], [40],

$$\boxed{\gamma = \arcsin \left(\frac{\boldsymbol{r} \cdot \boldsymbol{v}}{rv} \right)} \tag{7.80}$$

Since the range of the inverse sine function is $\left[-\frac{\pi}{2}, \frac{\pi}{2} \right]$, it follows that the domain of γ is [75, pp. 20–21]

$$\boxed{-\frac{\pi}{2} \leq \gamma \leq \frac{\pi}{2}} \tag{7.81}$$

Algorithm 54: `rv2fpa`
Flight path angle from position and velocity.

Inputs:
- $[\boldsymbol{r}]_{\mathcal{I}} \in \mathbb{R}^3$ — position expressed in an inertial frame $[\mathrm{m}]$
- $[\boldsymbol{v}]_{\mathcal{I}} \in \mathbb{R}^3$ — inertial velocity expressed in an inertial frame $[\mathrm{m/s}]$

Procedure:

$$\gamma = \arcsin\left(\frac{[\boldsymbol{r}]_{\mathcal{I}} \cdot [\boldsymbol{v}]_{\mathcal{I}}}{\|[\boldsymbol{r}]_{\mathcal{I}}\| \, \|[\boldsymbol{v}]_{\mathcal{I}}\|} \right)$$

return γ

Outputs:
- $\gamma \in \mathbb{R}$ — flight path angle $[\mathrm{rad}]$

Note:
- $-\frac{\pi}{2} \leq \gamma \leq \frac{\pi}{2}$

Test Cases:
- See Appendix A.5.1.

Relationship between flight path angle and specific angular momentum

Since β is the angle between $\boldsymbol{r}$ and $\boldsymbol{v}$, from the definition of the cross product (see Eq. (2.24) in Section 2.7) we have

$$|\boldsymbol{r} \times \boldsymbol{v}| = rv \sin \beta$$

Since $\boldsymbol{h} = \boldsymbol{r} \times \boldsymbol{v}$,

$$|\boldsymbol{h}| = h = rv \sin \beta$$

Substituting Eq. (7.79) into the equation above [13, p. 18],

$$h = rv \sin\left(\frac{\pi}{2} - \gamma\right)$$

$$\boxed{h = rv \cos \gamma} \tag{7.82}$$

From Fig. 7.8, we can see that for $\boldsymbol{r} \cdot \boldsymbol{v} > 0$, the flight path angle should be positive, while for $\boldsymbol{r} \cdot \boldsymbol{v} < 0$, the flight path angle should be negative. By simply solving Eq. (7.82) for $\cos \gamma$ and then taking the inverse cosine of both sides, we would get a sign error for when $\boldsymbol{r} \cdot \boldsymbol{v} < 0$, since the inverse cosine function can only produce positive numbers (its range is $[0, \pi]$). Therefore, we solve for γ piece-wise such that it always has the same sign as $\boldsymbol{r} \cdot \boldsymbol{v}$ [13, p. 18].

$$\boxed{\gamma = \begin{cases} \arccos\left(\dfrac{h}{rv}\right), & \boldsymbol{r} \cdot \boldsymbol{v} \geq 0 \\[2ex] -\arccos\left(\dfrac{h}{rv}\right), & \boldsymbol{r} \cdot \boldsymbol{v} < 0 \end{cases}} \tag{7.83}$$

Flight path angle from radial and along-track velocity components

In the previous section (Section 7.6.5), we discussed the inertial kinematics of a satellite expressed in the RSW frame. Specifically, we introduced the radial (v_R) and along-track (v_S) velocity components. These velocity components are shown visually in Fig. 7.9 together with the flight path angle. From Fig. 7.9, we can immediately see that
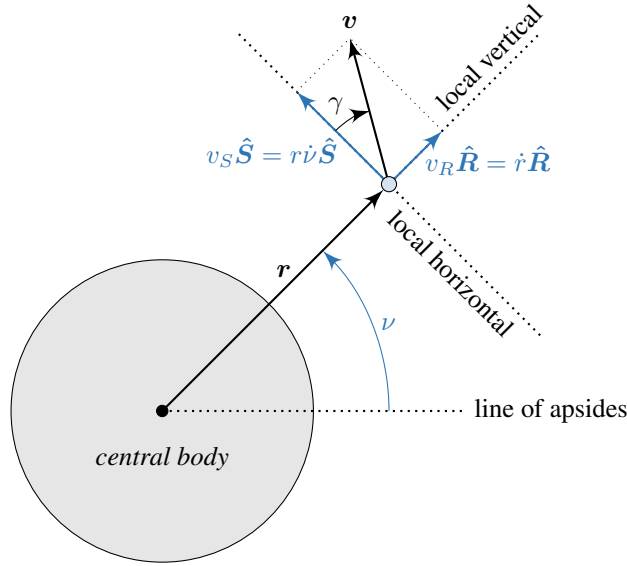
Figure 7.9: Radial and along-track velocity components defining the flight path angle [103, p. 25], [13, p. 32].

$$\boxed{\tan\gamma = \frac{v_R}{v_S}} \tag{7.84}$$

Substituting Eqs. (7.74) and (7.76) [23, pp. 77–79],

$$\tan\gamma = \left(\frac{\mu e \sin\nu}{h}\right)\left[\frac{h}{\mu(1 + e\cos\nu)}\right] = \frac{e\sin\nu}{1 + e\cos\nu}$$

Since the domain of the flight path angle is $\gamma \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$, and since the range of that inverse tangent function is also that same interval $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$, we can solve for γ directly [75, pp. 21-22].

$$\boxed{\gamma = \arctan\left(\frac{e\sin\nu}{1 + e\cos\nu}\right)} \tag{7.85}$$

We can obtain an alternative expression by substituting Eqs. (7.72) and (7.74) into Eq. (7.84) [.]

$$\boxed{\tan\gamma = \frac{\dot{r}}{r\dot{\nu}}} \tag{7.86}$$

Radial and along-track velocity components from the flight path angle

From Fig. 7.9, we can apply some simple trigonometry to find that

$$\boxed{v_R = v\sin\gamma} \tag{7.87}$$

$$\boxed{v_S = v\cos\gamma} \tag{7.88}$$

where $v = |\boldsymbol{v}|$ is the magnitude of the inertial velocity.

Flight path angle at the apsides

At both periapsis ($\nu = 0$) and apoapsis ($\nu = \pi$) (see Section 7.6.1), the flight path angle is 0.

$$\gamma_p = \gamma(\nu = 0) = \arctan\left(\frac{e \sin 0}{1 + e \cos 0}\right) = \arctan\left(\frac{0}{1+e}\right) = \arctan 0 = 0$$

$$\gamma_a = \gamma(\nu = \pi) = \arctan\left(\frac{e \sin \pi}{1 + e \cos \pi}\right) = \arctan\left(\frac{0}{1-e}\right) = \arctan 0 = 0$$

$$\boxed{\underbrace{\gamma_p}_{\text{periapsis}} = \underbrace{\gamma_a}_{\text{apoapsis}} = 0} \tag{7.89}$$

Alternatively, we could have used Eq. (7.86) and noted that the radial rate, $\dot{r}$, is 0 at both apsides (see Section 7.6.1). Since γ is 0 at the apsides, we know that $\beta = \pi/2$ at the apsides. Furthermore, since Eq. (7.78) gives

$$\boldsymbol{r} \cdot \boldsymbol{v} = rv \cos \beta$$

we know that if $\beta = \frac{\pi}{2}$ then $\boldsymbol{r} \cdot \boldsymbol{v} = 0$, implying that $\boldsymbol{r}$ and $\boldsymbol{v}$ are orthogonal.

$$\boxed{\text{The position, } \boldsymbol{r}, \text{ and inertial velocity, } \boldsymbol{v}, \text{ are orthogonal at the apsides.}}$$

Angular momentum at the apsides

Since the flight path angle is 0 at the apsides, we can find simple equations for the specific angular momentum. Let r_p and v_p be the position and inertial velocity magnitudes at periapsis. Similarly, let r_a and v_a be the position and inertial velocity magnitudes at apoapsis. The specific angular momentum magnitude at periapsis, h_p, is

$$h_p = r_p v_p \cos 0 = r_p v_p$$

The specific angular momentum magnitude at apoapsis, h_a, is

$$h_a = r_a v_a \cos 0 = r_a v_a$$

Since the specific angular momentum is constant for a Keplerian orbit, we have $h = h_p = h_a$, so [13, p. 27], [103, p. 24]

$$\boxed{h = r_p v_p = r_a v_a} \tag{7.90}$$

7.6.7   Specific Angular Momentum, Specific Mechanical Energy, and the Vis-Viva Equation

> Recall from Sections 7.3 and 7.4 that specific mechanical energy and specific angular momentum are conserved (i.e. constant) in a Keplerian orbit.

Specific angular momentum

Recall Eq. (7.28) defining the semiparameter of a Keplerian orbit.

$$p = \frac{h^2}{\mu} \tag{7.28}$$

Solving for the specific angular momentum, h,

$$\boxed{h = \sqrt{\mu p}} \tag{7.91}$$

Also recall from Eq. (7.36) that

$$p = a(1 - e^2)$$

Substituting this into Eq. (7.91) [13, pp. 24, 26, 28],

$$\boxed{h = \sqrt{\mu a(1 - e^2)}} \tag{7.92}$$

Specific mechanical energy

Recall Eq. (6.26) from Section 6.6 defining the specific mechanical energy of an orbiting body.

$$\mathcal{E} = \frac{v^2}{2} - \frac{\mu}{r} \tag{6.26}$$

Since $\mathcal{E}$ is a constant, we can write it in terms of the position and inertial velocity magnitudes at periapsis (r_p and v_p, respectively).

$$\mathcal{E} = \frac{v_p^2}{2} - \frac{\mu}{r_p}$$

Also recall from Eq. (7.90) that the specific angular momentum can be written in terms of r_p and v_p as

$$h = r_p v_p$$

Solving for v_p,

$$v_p = \frac{h}{r_p}$$

Substituting this into Eq. (6.26),

$$\mathcal{E} = \frac{h^2}{2r_p^2} - \frac{\mu}{r_p}$$

From Eq. (7.39) we know that

$$r_p = a(1 - e)$$

Substituting this into our expression for $\mathcal{E}$,

$$\mathcal{E} = \frac{h^2}{2a^2(1 - e)^2} - \frac{\mu}{a(1 - e)} \tag{7.93}$$

Squaring Eq. (7.92), we also know that

$$h^2 = \mu a(1 - e^2) \tag{7.94}$$

Substituting this into Eq. (7.93) and simplifying [13, pp. 27–28], [75, p. 19],

$$\mathcal{E} = \frac{\mu a(1 - e^2)}{2a^2(1 - e)^2} - \frac{\mu}{a(1 - e)} = \frac{\mu(1 - e)(1 + e)}{2a(1 - e)^2} - \frac{\mu}{a(1 - e)} = \frac{\mu(1 + e)}{2a(1 - e)} - \frac{\mu}{a(1 - e)}$$

$$= \frac{\mu(1 + e)}{2a(1 - e)} - \frac{2\mu}{2a(1 - e)} = \frac{\mu(1 + e) - 2\mu}{2a(1 - e)} = \frac{\mu + \mu e - 2\mu}{2a(1 - e)} = \frac{-\mu + \mu e}{2a(1 - e)} = \frac{-\mu(1 - e)}{2a(1 - e)}$$

$$\boxed{\mathcal{E} = -\frac{\mu}{2a}} \tag{7.95}$$

We can obtain yet another alternative expression for $\mathcal{E}$ by solving Eq. (7.94) for a and substituting it into Eq. (7.95) [23, p. 81].

$$a = \frac{h^2}{\mu(1 - e^2)} \quad \rightarrow \quad \mathcal{E} = -\frac{\mu}{2}\left[\frac{\mu(1 - e^2)}{h^2}\right]$$

$$\mathcal{E} = \frac{\mu^2(e^2-1)}{2h^2}$$ (7.96)

We can also obtain another expression for the eccentricity, e, by first recalling Eq. (7.38) from Section 7.6.1.

$$e = \sqrt{1 - \frac{p}{a}}$$ (7.38)

Solving Eq. (7.95) for a and substituting it into Eq. (**??**),

$$a = -\frac{\mu}{2\mathcal{E}} \quad \rightarrow \quad e = \sqrt{1 + \frac{2\mathcal{E}p}{\mu}}$$

Next, substituting Eq. (7.28) [13, pp. 28–29],

$$e = \sqrt{1 + \frac{2\mathcal{E}(h^2/\mu)}{\mu}}$$

$$e = \sqrt{1 + \frac{2\mathcal{E}h^2}{\mu^2}}$$ (7.97)

Vis-viva equation

Substituting Eq. (7.95) back into Eq. (6.26),

$$-\frac{\mu}{2a} = \frac{v^2}{2} - \frac{\mu}{r}$$

Solving for v,

$$\frac{v^2}{2} = \frac{\mu}{r} - \frac{\mu}{2a} \quad \rightarrow \quad v^2 = \frac{2\mu}{r} - \frac{\mu}{a} = \mu\left(\frac{2}{r} - \frac{1}{a}\right)$$

$$v = \sqrt{\mu\left(\frac{2}{r} - \frac{1}{a}\right)}$$ (7.98)

Eq. (7.98) is known as the **vis-viva**[5] **equation** [75, pp. 19-20].

Recall that r varies with the true anomaly, ν, as governed by the trajectory equation. We can rewrite Eq. (7.98) by substituting the alternate form of the trajectory equation given by Eq. (7.46) [103, p. 32].

$$r = \frac{a(1-e^2)}{1 + e\cos\nu} \quad \rightarrow \quad v = \sqrt{\mu\left[\frac{2(1+e\cos\nu)}{a(1-e^2)} - \frac{1}{a}\right]}$$

Simplifying,

$$v = \sqrt{\mu\left[\frac{2(1+e\cos\nu)}{a(1-e^2)} - \frac{(1-e^2)}{a(1-e^2)}\right]} = \sqrt{\mu\left[\frac{2(1+e\cos\nu)-(1-e^2)}{a(1-e^2)}\right]} = \sqrt{\mu\left[\frac{2+2e\cos\nu-1+e^2}{a(1-e^2)}\right]}$$

$$= \sqrt{\mu\left[\frac{e^2+2e\cos\nu+1}{a(1-e^2)}\right]}$$

---

[5] Latin for "living force" [104]

$$v = \sqrt{\frac{\mu(e^2 + 2e\cos\nu + 1)}{a(1 - e^2)}} \qquad (7.99)$$

We can also solve the vis-viva equation (given by Eq. (7.98)) for r and a. First, let's solve for r

$$v^2 = \mu\left(\frac{2}{r} - \frac{1}{a}\right) \quad \rightarrow \quad \frac{v^2}{\mu} = \frac{2}{r} - \frac{1}{a} \quad \rightarrow \quad \frac{2}{r} = \frac{v^2}{\mu} + \frac{1}{a} = \frac{av^2 + \mu}{\mu a} \quad \rightarrow \quad \frac{r}{2} = \frac{\mu a}{\mu + av^2}$$

$$r = \frac{2\mu a}{\mu + av^2} \qquad (7.100)$$

Next, solving instead for a,

$$v^2 = \mu\left(\frac{2}{r} - \frac{1}{a}\right) \quad \rightarrow \quad \frac{v^2}{\mu} = \frac{2}{r} - \frac{1}{a} \quad \rightarrow \quad \frac{1}{a} = \frac{2}{r} - \frac{v^2}{\mu} = \frac{2\mu - rv^2}{\mu}$$

$$a = \frac{\mu}{2\mu - rv^2} \qquad (7.101)$$

TODO: unit test for v from conic section parameters TODO: algorithms/matlab functions?

7.7   Elliptical Orbits

7.7.1   Geometry

$$b = a\sqrt{1 - e^2} \qquad (7.102)$$

$a^2 = b^2 + c^2$ because of conic section definition (p. 15 of vallado) TODO: derive Eq. (7.102)
TODO: pp. 85-90 curtis TODO: around p. 29 in vallado

7.7.2   Kepler's First Law: Elliptical Orbits

Kepler's first law states that the orbit of each planet is an ellipse with the Sun at one focus [103, p. 10], [13, p. 2], [43]. While this law also has an observational aspect (we must actually observe the planets orbiting the Sun with closed orbits), we essentially proved its mathematical aspect through our derivation (Section **??**), solution (Section **??**), and geometric interpretation (Section 7.6) of the Fundamental Orbital Differential Equation (FODE).

7.7.3   Kepler's Second Law: Equal Areas in Equal Time

Consider an elliptical orbit where at time t we have true anomaly ν and orbital radius r. An instant later, at time $t + \Delta t$, the orbital radius is still (approximately) r (at least in the limit as $\Delta t \to 0$, but the true anomaly has advanced by $\Delta\nu$. Since the area of a sector of a circle of radius r subtended by an angle θ is $A = \theta r^2/2$, we have that the orbit swept out an incremental area

$$\Delta A = \frac{1}{2}r^2\Delta\nu$$

in time Δt. Dividing both sides by Δt,

$$\frac{\Delta A}{\Delta t} = \frac{r^2}{2}\left(\frac{\Delta\nu}{\Delta t}\right)$$

Taking the limit as $\Delta t \to 0$, we obtain the derivative

$$\frac{dA}{dt} = \frac{r^2}{2}\left(\frac{d\nu}{dt}\right) \qquad (7.103)$$

Writing this using Newton's notation,

$$\dot{A} = \frac{r^2 \dot{\nu}}{2}$$

Recall Eq. (7.69), repeated below for convenience.

$$\dot{\nu} = \frac{h}{r^2} \tag{7.69}$$

Substituting this into our expression for $\dot{A}$,

$$\dot{A} = \frac{r^2}{2}\left(\frac{h}{r^2}\right) = \frac{h}{2}$$

Since h is constant for a Keplerian orbit (see Section 7.4), we have

$$\boxed{\dot{A} = \frac{h}{2} = \text{constant}} \tag{7.104}$$

Equation (7.104) proves **Kepler's second law**, which states that the line joining the plane to the Sun (i.e. the position vector of the planet) sweeps out equal areas in equal time intervals [103, pp. 29–30], [13, pp. 2, 31–33], [23, p. 74], [75, pp. 22-23], [43].

TODO: do actually geometry with a figure; see bmw and curtis references TODO: p. 748 in james stewart

7.7.4 Kepler's Third Law: Orbital Period

see p. 29 in vallado TODO: use references from previous section

We know that the area of an ellipse is

$$A = \pi a b$$

From Eq. (7.102), we also know that

$$b = a\sqrt{1 - e^2}$$

so we have

$$A = \pi a^2 \sqrt{1 - e^2} \tag{7.105}$$

The **orbital period**, T, is the amount of time it takes to complete one elliptical[6] orbit. The area swept out by the position vector over the course of one complete elliptical orbit is

$$A = \int_0^T \dot{A}\,dt$$

where $\dot{A}$ is given by Eq. (7.104) from the previous section. Since we showed that $\dot{A} = $ constant (i.e. Kepler's second law), we can pull it out from under the integrand and find

$$A = \dot{A}\int_0^T dt$$
$$= \dot{A}T$$

Substituting Eq. (7.104) and solving for T,

$$A = \frac{hT}{2} \quad \rightarrow \quad T = \frac{2A}{h}$$

---

[6] Note that a circle is a special case of an ellipse.

Substituting Eq. (**??**) for h,

$$T = \frac{2A}{\sqrt{\mu a(1 - e^2)}}$$

(7.106)

Substituting Eq. (7.105) into Eq. (**??**),

$$T = \frac{2\pi a^2 \sqrt{1 - e^2}}{\sqrt{\mu a(1 - e^2)}} = \frac{2\pi \sqrt{a^4(1 - e^2)}}{\sqrt{\mu a(1 - e^2)}} = 2\pi \sqrt{\frac{a^4(1 - e^2)}{\mu a(1 - e^2)}}$$

$$\boxed{T = 2\pi \sqrt{\frac{a^3}{\mu}}}$$

(7.107)

Equation (7.107) proves **Kepler's third law**, which states that the square of a planet's orbital period around the Sun is proportional to the cube of the semi-major axis of its orbit [43], [13, p. 2].

$$T^2 \propto a^3$$

In a more general sense, we can simply neglect the words "around the Sun" to note that Kepler's third law is valid for any elliptical orbit about any primary focus. Also, note the remarkable result that the period of an elliptical orbit is independent of its eccentricity.

We can also rearrange Eq. (7.107) to solve for the semimajor axis, a.

$$T^2 = \frac{4\pi^2 a^3}{\mu} \quad \rightarrow \quad a^3 = \frac{\mu T^2}{4\pi^2} = \mu \left(\frac{T}{2\pi}\right)^2$$

$$\boxed{a = \left[\mu \left(\frac{T}{2\pi}\right)^2\right]^{1/3}}$$

(7.108)

Equations (7.107) and (7.108) are formalized as Algorithms 55 and 56 below, respectively.

Algorithm 55: `a2T`
Orbital period from semi-major axis.

Inputs:
- $a \in \mathbb{R}$ - semi-major axis $[\text{m}]$
- $\mu \in \mathbb{R}$ - standard gravitational parameter $[\text{m}^3/\text{s}^2]$

Procedure:

$$T = 2\pi \sqrt{\frac{a^3}{\mu}}$$

return T

Outputs:
- $T \in \mathbb{R}$ - orbital period $[\text{s}]$

Note:
- This algorithm is only valid for circular ($e = 0$) and elliptical ($0 < e < 1$) orbits.

Test Cases:
- See Appendix A.5.1.

Algorithm 56: `T2a`
Semi-major axis from orbital period.

Inputs:
- $T \in \mathbb{R}$ - orbital period $[\text{s}]$
- $\mu \in \mathbb{R}$ - standard gravitational parameter $[\text{m}^3/\text{s}^2]$

Procedure:

$$a = \left[\mu \left(\frac{T}{2\pi} \right)^2 \right]^{1/3}$$

   return a

Outputs:
- $a \in \mathbb{R}$ - semi-major axis $[\text{m}]$

Note:
- This algorithm is only valid for circular $(e = 0)$ and elliptical $(0 < e < 1)$ orbits.

Test Cases:
- See Appendix A.5.1.

7.7.5   Mean Motion

In a single elliptical orbit, the true anomaly, ν, moves through 2π radians in one orbit period, T. However, as the satellite moves along its orbit, its true anomaly does not change at a constant rate. Recall Eq. (7.69) from Section 7.6.5, repeated below for convenience.

$$\dot{\nu} = \frac{h}{r^2} \tag{7.69}$$

While h is constant for a Keplerian orbit, the radial distance, r, is *not* constant for elliptical orbits, so $\dot{\nu}$ is not constant either.

$$\frac{d}{dt}(\dot{\nu}) \neq 0 \quad \rightarrow \quad \dot{\nu} \neq \text{constant}$$

However, it is useful to define an angular rate that *is* constant for any elliptical orbit. We define the **mean motion**, n, as the *constant* angular rate that results in one complete orbit (2π radians) in one orbital period (T).

$$\boxed{n = \frac{2\pi}{T}} \tag{7.109}$$

We will make more use of the mean motion in subsequent sections, but for the remainder of this section, we will define its relationships with the orbital period and semi-major axis. First, by rearranging Eq. (7.109), we can get T in terms of n.

$$\boxed{T = \frac{2\pi}{n}} \tag{7.110}$$

7.8   True, Mean, and Eccentric Anomalies

Mean Anomaly ↔ Eccentric Anomaly

To obtain the mean anomaly from the eccentric anomaly, we can just use Kepler's equation, given by Eq. (**??**).

Algorithm 57: E2M
Mean anomaly from eccentric anomaly.

Inputs:
- $E \in \mathbb{R}$ - eccentric anomaly [rad]
- $e \in \mathbb{R}$ - eccentricity

Procedure:

$$M = E - e \sin E$$
return M

Outputs:
- $M \in \mathbb{R}$ - mean anomaly [rad]

Note:
- $0 \leq e < 1$
- $0 \leq E \leq 2\pi$
- $0 \leq M \leq 2\pi$

Going in the opposite direction is trickier, since there is no general closed form solution for E. However, we first note that $\sin n\pi = 0$. Since $E \in [0, 2\pi)$, we have that

$$M(0) = 0 - e \sin 0 = 0$$
$$M(\pi) = \pi - e \sin \pi = \pi$$

Next, let's rearrange Kepler's equation and define the function $f(E)$:

$$\underbrace{E - e \sin E - M}_{f(E)} = 0$$

Thus, E is just the root of the function $f(E)$. We can solve this equation using Newton's method, where the estimate of the root is updated as

$$x_{i+1} = x_i + \frac{f(x_i)}{f'(x_i)}$$

See [**kis_newtons_method**] for more information on Newton's method and [**kis_newtons_method_newtons_method**] for a general implementation for Newton's method.

In this case, we update the current estimate for the eccentric anomaly as

$$E_{i+1} = E_i + \frac{f(E_i)}{f'(E_i)}$$

where the derivative of $f(E)$ is

$$f'(E) = 1 - e \cos E$$

Thus,

$$E_{i+1} = E_i + \frac{E_i - e \sin E_i}{1 - e \cos E_i}$$

Note that $E, M \in [0, 2\pi)$. To accelerate convergence of the algorithm, we choose the initial guess as follows [103, pp. 65, 75]:

$$E_0 = \begin{cases} M + e, & M < \pi \\ M - e, & M > \pi \end{cases}$$

Algorithm 58: M2E
Eccentric anomaly from mean anomaly.

Inputs:
  - $M \in \mathbb{R}$ - mean anomaly $[\mathrm{rad}]$
  - $e \in \mathbb{R}$ - eccentricity

Procedure:
  1. Set the tolerance (TOL) and maximum number of iterations ($k_{\max}$).

$$\mathrm{TOL} = 10^{-12}$$
$$k_{\max} = 200$$

  2. Return known solutions.

> **if** $M = 0$ **or** $M = \pi$
> > $E = M$
> >   **return**
> **end**

  3. Wraps M to the interval $[0, 2\pi)$ if $M \notin [0, 2\pi)$.

> **if** $M < 0$ **or** $M \geq 2\pi$
> > $M = \mathrm{mod}\,(M, 2\pi)$
> **end**

  4. Set initial guess for the eccentric anomaly based on the value of M.

> **if** $M < \pi$
> > $E_{\mathrm{old}} = M + e$
> **else**
> > $E_{\mathrm{old}} = M - e$
> **end**

  5. Initialize E_{new} so its scope will not be limited to within the while loop.

$$E_{\mathrm{new}} = 0$$

  6. Initialize the error so that the loop will be entered.

$$\varepsilon = (2)(\mathrm{TOL})$$

  7. Find the eccentric anomaly using Newton's method.

$$k = 1$$

while $(\varepsilon > \text{TOL})$ **and** $(k < k_{\max})$

    (a) Update eccentric anomaly estimate.

$$E_{\text{new}} = E_{\text{old}} + \frac{E_{\text{old}} - e \sin E_{\text{old}}}{1 - e \cos E_{\text{old}}}$$

    (b) Calculate error.

$$\varepsilon = |E_{\text{new}} - E_{\text{old}}|$$

    (c) Store the current eccentric anomaly estimate for the next itera-
tion.

$$E_{\text{old}} = E_{\text{new}}$$

    (d) Increment loop index.

$$k = k + 1$$

end

8. Store the final solution for the eccentric anomaly.

$$E = E_{\text{new}}$$

9. Return the result.

    return E

Outputs:

- $E \in \mathbb{R}$ - eccentric anomaly $[\text{rad}]$

Note:

- $0 \leq e < 1$
- $0 \leq M \leq 2\pi$
- $0 \leq E \leq 2\pi$

Eccentric Anomaly $\leftrightarrow$ True Anomaly

To solve for the true anomaly, ν, from the eccentric anomaly, E, we know that [103, pp. 47–48]

$$\sin \nu = \frac{\sqrt{1 - e^2} \sin E}{1 - \cos E} \tag{7.111}$$

$$\cos \nu = \frac{\cos E - e}{1 - e \cos E} \tag{7.112}$$

Dividing Eq. (7.115) by Eq. (7.116),

$$\frac{\sin \nu}{\cos \nu} = \left(\frac{\sqrt{1 - e^2} \sin E}{1 - \cos E} \right) \left(\frac{1 - e \cos E}{\cos E - e} \right) \quad \rightarrow \quad \tan \nu = \frac{\sqrt{1 - e^2} \sin E}{\cos E - e}$$

To solve for ν in a way that ensures that ν is returned in the correct quadrant, we can use the four-quadrant (i.e. two argument) inverse tangent ($\arctan 2$).

$$\boxed{\nu = \arctan 2 \left(\sqrt{1 - e^2} \sin E, \ \cos E - e \right)} \tag{7.113}$$

Alternatively, we can use [103, p. 48]

$$\nu = 2\arctan\left(\sqrt{\frac{1+e}{1-e}}\tan\frac{E}{2}\right) \tag{7.114}$$

However, note that the computational implementations of $\arctan$ and $\arctan2$ typically return angles in the interval $[-\pi/2, \pi/2]$, while we define ν such that $\nu \in [0, 2\pi)$. Therefore, we must wrap the result to 2π using the modulo operation.

Algorithm 59: `E2nu`
True anomaly from eccentric anomaly.

Inputs:
- $E \in \mathbb{R}$ - eccentric anomaly [rad]
- $e \in \mathbb{R}$ - eccentricity

Procedure:

$$\nu = \mathrm{mod}\left[2\arctan\left(\sqrt{\frac{1+e}{1-e}}\tan\frac{E}{2}\right), 2\pi\right]$$

   return ν

Outputs:
- $\nu \in \mathbb{R}$ - true anomaly [rad]

Note:
- $0 \le e < 1$
- $0 \le E \le 2\pi$
- $0 \le \nu \le 2\pi$

To solve for the eccentric anomaly, E, from the true anomaly, ν, we know that [103, p. 47]

$$\sin E = \frac{\sqrt{1-e^2}\sin\nu}{1+e\cos\nu} \tag{7.115}$$

$$\cos E = \frac{e+\cos\nu}{1+e\cos\nu} \tag{7.116}$$

Dividing Eq. (7.115) by Eq. (7.116),

$$\frac{\sin E}{\cos E} = \left(\frac{\sqrt{1-e^2}\sin\nu}{1+e\cos\nu}\right)\left(\frac{1+e\cos\nu}{e+\cos\nu}\right) \quad \rightarrow \quad \tan E = \frac{\sqrt{1-e^2}\sin\nu}{e+\cos\nu}$$

To solve for E in a way that ensures that E is returned in the correct quadrant, we can use the four-quadrant (i.e. two argument) inverse tangent ($\arctan2$).

$$E = \arctan2\left(\sqrt{1-e^2}\sin\nu,\ e+\cos\nu\right) \tag{7.117}$$

Alternatively, we can use [103, p. 48]

$$E = 2\arctan\left(\sqrt{\frac{1-e}{1+e}}\tan\frac{\nu}{2}\right) \tag{7.118}$$

Just like before, we need to wrap the results to 2π using the modulo operation since we define E such that $E \in [0, 2\pi)$.

Algorithm 60: `nu2E`
Eccentric anomaly from true anomaly.

Inputs:
- $\nu \in \mathbb{R}$ - true anomaly $[\mathrm{rad}]$
- $e \in \mathbb{R}$ - eccentricity

Procedure:

$$E = \mathrm{mod}\left[2\arctan\left(\sqrt{\frac{1-e}{1+e}}\tan\frac{\nu}{2}\right), 2\pi\right]$$

   return E

Outputs:
- $E \in \mathbb{R}$ - eccentric anomaly $[\mathrm{rad}]$

Note:
- $0 \le e < 1$
- $0 \le \nu \le 2\pi$
- $0 \le E \le 2\pi$

Mean Anomaly $\leftrightarrow$ True Anomaly

To calculate the mean anomaly given the true anomaly, and vice-versa, we simply "combine" the algorithms presented in Sections **??** and **??**.

Algorithm 61: `nu2M`
Mean anomaly from true anomaly.

Inputs:
- $\nu \in \mathbb{R}$ - true anomaly $[\mathrm{rad}]$
- $e \in \mathbb{R}$ - eccentricity

Procedure:
1. Eccentric anomaly $[\mathrm{rad}]$ (Algorithm 60).

   $$E = \mathtt{nu2E}(\nu, e)$$

2. Mean anomaly $[\mathrm{rad}]$ (Algorithm 57).

   $$M = \mathtt{E2M}(E, e)$$

3. Return the result.

   return M

Outputs:
  • $M \in \mathbb{R}$ - mean anomaly [rad]

Note:
  • $0 \le e < 1$
  • $0 \le \nu \le 2\pi$
  • $0 \le M \le 2\pi$

Algorithm 62: `M2nu`
True anomaly from mean anomaly.

Inputs:
  • $M \in \mathbb{R}$ - mean anomaly [rad]
  • $e \in \mathbb{R}$ - eccentricity

Procedure:
  1. Eccentric anomaly [rad] (Algorithm 58).

$$E = \mathtt{M2E}(M, e)$$

  2. True anomaly [rad] (Algorithm 59).

$$\nu = \mathtt{E2nu}(E, e)$$

  3. Return the result.

return ν

Outputs:
  • $\nu \in \mathbb{R}$ - true anomaly [rad]

Note:
  • $0 \le e < 1$
  • $0 \le M \le 2\pi$
  • $0 \le \nu \le 2\pi$

7.8.1   Radial, Transverse, and True Anomaly Rates

TODO: start from section 9.6.5 and get purely in terms of orbital elements

TODO: put after specific energy/specific ang. momentum section, use eq. 1-32 from pg 32 of vallado to get velocity, use trajectory equation for position, do we need this though?

TODO: radial/transverse rates: pp. 77-78 curtis, inside cover vallado (plus elsewhere in vallado), check BMW See page 399 of BMW for elliptical case p. 29 of vallado

add notes on previous sections about semiparameter in terms of a and e, note the special case for parabolas but write the equations in general besides that

page 24–25 for $h = r^2 \dot{\nu}$ using FPA, also done around p. 77-78 of curtis

TODO: diagram, already in tikz diagrams I htink, but cite p. 25 of vallado for inspiration and p. 32 of bmw

TODO

Angle between $\hat{S}$ and v is the flight path angle:

$$\therefore v_S = \frac{h}{r} = \frac{\mu(1 + e\cos\nu)}{\sqrt{a\mu(1 - e^2)}}$$

In the radial direction,

$$v_R = \dot{r} = \frac{dr}{dt} = \frac{d}{dt}\left[\frac{h^2}{\mu}\frac{1}{1 + e\cos\mu}\right] = \frac{h^2}{\mu}\frac{d}{dt}\left[\frac{1}{1 + e\cos\mu}\right] = \frac{h^2}{\mu}\left[\frac{e\sin\nu}{(1 + e\cos\nu)^2}\frac{h}{r^2}\right]$$

$$\therefore v_R = \frac{\mu e\sin\nu}{\sqrt{a\mu(1 - e^2)}}$$

TODO: lots of stuff starting on p.15 of vallado TODO: p. 31 of bmw also has useful eccentriity for ellipse TODO: geometry of flight path angle in vallado for elliptical orbits (p. 19 in vallado) TODO: sign of fpa for elliptical (p. 99ish from Tewari)

7.9   Circular Orbits

TODO: circular velocity (p. 33 of vallado)

7.10   Parabolic Orbits

TODO: escape velocity TODO: fpa on p. 96 of curtis TODO: energy and escape velocity on p. 33 of vallado TODO: v-infinity (p. 27 of vallado) TODO: velocity (p. 33 of vallado)

TODO: orbital altitude

8

Orbits in Three Dimensions

TODO: transformation from PQW to RSW TODO: picture from tikz true anomaly

8.1 Planet-Centered Inertial (PCI) Frames
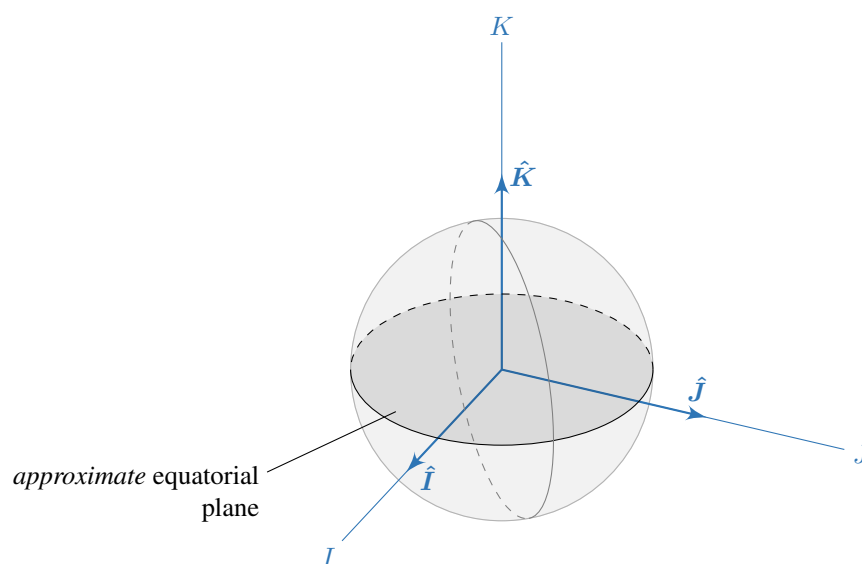
Figure 8.1: Planet-centered inertial (PCI) coordinate frame.

   A **planet-centered inertial (PCI) coordinate frame** is an inertial frame used to describe orbital motion about a planetary body. It has the following characteristics:

1. Its origin is located at the center of the planetary body.
2. It has coordinate axes I, J, and K, defined by the basis vectors $\hat{I}$, $\hat{J}$, and $\hat{K}$.
   (a) The I axis (1st axis) points in the direction of the **vernal equinox** (i.e. in the direction of the **first point of Aries**).

(b) The J axis (2nd axis) is normal to the I axis, such that the IJ-plane is *approximately* aligned with the equatorial plane.

(c) The K axis (3rd axis) completes the right-handed triad, and is in the *approximate* direction of the north celestial pole.

3. Its coordinate axes are fixed with respect to inertial space; simply put, it remains fixed with respect to the distant stars.

Figure 8.1 provides an illustration of a planet-centered inertial frame.

Since planetary bodies undergo precession and nutation, the PCI frame will never have its IJ-plane exactly aligned with the true equatorial plane. In general, PCI frames are defined at a certain epoch, and the orientation of the planetary body is defined as a rotation with respect to the PCI frame. This will be covered in significant detail in Chapter 10. In this chapter, we simply need to understand what a PCI frame is.

> As mentioned previously, PCI frames are specific to a certain epoch (i.e. point in time; see Section 9.1 for more details). Therefore, a given planetary body will have many different PCI frames.

8.2   Orbital State Vector

In Chapters 6 and 7, we discussed some fundamentals of orbital mechanics in terms of generic position ($\boldsymbol{r}$) and inertial velocity ($\boldsymbol{v}$) vectors, expressed in any inertial frame[1]. Now, as we begin to orbits in three dimensions, it is critical to note which inertial frame we are using. In this chapter, we will use a generic PCI frame, but note that in most applications, this is replaced with a very specific coordinate frame unique to the planetary body in question.

Consider the position of a satellite expressed in a PCI frame. Writing out the coordinate vector,

$$[\boldsymbol{r}]_{\text{pci}} = \begin{bmatrix} r_I \\ r_J \\ r_K \end{bmatrix} \tag{8.1}$$

Also consider the velocity of the satellite relative to and expressed in the PCI frame. Writing out the coordinate vector,

$$[^{\text{pci}}\boldsymbol{v}]_{\text{pci}} = \begin{bmatrix} v_I \\ v_J \\ v_K \end{bmatrix} \tag{8.2}$$

The **orbital state vector**, $\boldsymbol{X}$, of a satellite about a planetary body is comprised of the position and PCI velocity of the satellite expressed in a PCI frame. Since the orbital state vector will be specific to the PCI frame used, we must label the frame. For the purpose of this definition, we just label it with the generic PCI label. In practice, this label will be replaced with the specific PCI frame used.

$$[\boldsymbol{X}]_{\text{pci}} = \begin{bmatrix} [\boldsymbol{r}]_{\text{pci}} \\ [^{\text{pci}}\boldsymbol{v}]_{\text{pci}} \end{bmatrix} = \begin{bmatrix} r_I \\ r_J \\ r_K \\ v_I \\ v_J \\ v_K \end{bmatrix} \tag{8.3}$$

---

[1]   Note that we also used a simplified notation, discussed in Sections **??** and 6.1.

8.3 Perifocal (PQW) Frame in Three Dimensions

8.4 RSW Frame in Three Dimensions

> See Section 7.6.5 for an in-depth definition and discussion of the RSW frame.

In Section 7.6.5, we defined a rotation matrix between the RSW frame (which is based on the instantaneous orbital state of the satellite) and the PQW frame, which is used to define the orbit geometry in the orbital plane. While the PQW frame *is* an inertial frame, it is specific to a given orbit. However, a PCI frame for a planetary body is completely independent of any orbit; it is only dependent on the orientation of a planet at a certain point in time. In this section, we develop the rotation matrix between a PCI frame and the PQW frame.

Recall Eqs. (7.53), (7.54), and (7.55) from Section 7.6.5 (repeated below for convenience), defining $\hat{R}$, $\hat{W}$, and $\hat{S}$, respectively.

$$\hat{R} = \frac{r}{r} \tag{7.53}$$

$$\hat{W} = \frac{r \times v}{|r \times v|} = \frac{\mathbf{h}}{h} = \hat{h} \tag{7.54}$$

$$\hat{S} = \hat{W} \times \hat{R} \tag{7.55}$$

Since v is an arbitrary inertial velocity vector, we can replace it with the velocity relative to the PCI frame, which is a more specific inertial velocity vector. Then $\hat{W}$ becomes

$$\hat{W} = \frac{r \times {}^{\mathrm{pci}}v}{|r \times {}^{\mathrm{pci}}v|} = \frac{{}^{\mathrm{pci}}h}{\left|{}^{\mathrm{pci}}h\right|}$$

The PCI specific angular momentum vector expressed in the PCI frame is just

$$\boxed{\left[{}^{\mathrm{pci}}h\right]_{\mathrm{pci}} = [\hat{r}]_{\mathrm{pci}} \times \left[{}^{\mathrm{pci}}v\right]_{\mathrm{pci}}} \tag{8.4}$$

Expressing the basis vectors of the RSW frame ($\hat{R}$, $\hat{S}$, $\hat{W}$) in the PCI frame,

$$\boxed{\left[\hat{R}\right]_{\mathrm{pci}} = \frac{[r]_{\mathrm{pci}}}{\|[r]_{\mathrm{pci}}\|}} \tag{8.5}$$

$$\boxed{\left[\hat{W}\right]_{\mathrm{pci}} = \frac{\left[{}^{\mathrm{pci}}h\right]_{\mathrm{pci}}}{\left\|\left[{}^{\mathrm{pci}}h\right]_{\mathrm{pci}}\right\|}} \tag{8.6}$$

$$\boxed{\left[\hat{S}\right]_{\mathrm{pci}} = \left[\hat{W}\right]_{\mathrm{pci}} \times \left[\hat{R}\right]_{\mathrm{pci}}} \tag{8.7}$$

Recall from Section 2.5.2 that we can construct a rotation matrix from one frame to another using the basis vectors of one frame expressed in the other frame. Specifically, Eq. (2.18) gives

$$\mathbf{R}_{\mathcal{B}\to\mathcal{A}} = \left([\hat{a}]_{\mathcal{A}} \quad [\hat{b}]_{\mathcal{A}} \quad [\hat{c}]_{\mathcal{A}}\right)$$

where $\left(\hat{a}, \hat{b}, \hat{c}\right)$ is the physical ordered basis defining frame $\mathcal{B}$. In this case, the origin frame ($\mathcal{B}$) is the RSW frame, while the destination frame ($\mathcal{A}$) is the PCI frame.

$$\boxed{\mathbf{R}_{\mathrm{rsw}\to\mathrm{pci}} = \left(\left[\hat{R}\right]_{\mathrm{pci}} \quad \left[\hat{S}\right]_{\mathrm{pci}} \quad \left[\hat{W}\right]_{\mathrm{pci}}\right)} \tag{8.8}$$

Since rotation matrices are orthogonal, the inverse rotation matrix (i.e. from PCI to RSW) is simply the transpose of the original rotation matrix (i.e. from RSW to PCI). Thus [103, p. 164],

$$
\mathbf{R}_{\text{pci}\to\text{rsw}} = \mathbf{R}^{T}_{\text{rsw}\to\text{pci}} = \left(\left[\hat{\boldsymbol{R}} \right]_{\text{pci}} \quad \left[\hat{\boldsymbol{S}} \right]_{\text{pci}} \quad \left[\hat{\boldsymbol{W}} \right]_{\text{pci}} \right)^{T} = \begin{pmatrix} \left[\hat{\boldsymbol{R}} \right]^{T}_{\text{pci}} \\ \left[\hat{\boldsymbol{S}} \right]^{T}_{\text{pci}} \\ \left[\hat{\boldsymbol{W}} \right]^{T}_{\text{pci}} \end{pmatrix}
\tag{8.9}
$$

Equations (**??**) and (8.9) are formalized as Algorithms 63 and 64 below.

Algorithm 63: `rot_rsw2pci`

Rotation matrix from Hill/orbital (RSW) frame to Earth-centered inertial (PCI) frame.

Inputs:
- $[\boldsymbol{r}]_{\text{pci}} \in \mathbb{R}^3$ - position expressed in PCI frame
- $\left[^{\text{pci}}\boldsymbol{v} \right]_{\text{pci}} \in \mathbb{R}^3$ - PCI velocity expressed in PCI frame

Procedure:
1. PCI specific angular momentum vector expressed in the PCI frame.

$$
\left[^{\text{pci}}\boldsymbol{h} \right]_{\text{pci}} = [\hat{\boldsymbol{r}}]_{\text{pci}} \times [^{\text{pci}}\boldsymbol{v}]_{\text{pci}}
$$

2. $\hat{\boldsymbol{R}}$, $\hat{\boldsymbol{S}}$, and $\hat{\boldsymbol{W}}$ basis vectors expressed in the PCI frame.

$$
\left[\hat{\boldsymbol{R}} \right]_{\text{pci}} = \frac{[\boldsymbol{r}]_{\text{pci}}}{\|[\boldsymbol{r}]_{\text{pci}}\|}
$$

$$
\left[\hat{\boldsymbol{W}} \right]_{\text{pci}} = \frac{\left[^{\text{pci}}\boldsymbol{h} \right]_{\text{pci}}}{\left\| \left[^{\text{pci}}\boldsymbol{h} \right]_{\text{pci}} \right\|}
$$

$$
\left[\hat{\boldsymbol{S}} \right]_{\text{pci}} = \left[\hat{\boldsymbol{W}} \right]_{\text{pci}} \times \left[\hat{\boldsymbol{R}} \right]_{\text{pci}}
$$

3. Rotation matrix from RSW frame to PCI frame.

$$
\mathbf{R}_{\text{rsw}\to\text{pci}} = \left(\left[\hat{\boldsymbol{R}} \right]_{\text{pci}} \quad \left[\hat{\boldsymbol{S}} \right]_{\text{pci}} \quad \left[\hat{\boldsymbol{W}} \right]_{\text{pci}} \right)
$$

4. Return the result.

$$
\textbf{return } \mathbf{R}_{\text{rsw}\to\text{pci}}
$$

Outputs:
- $\mathbf{R}_{\text{rsw}\to\text{pci}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from RSW frame to PCI frame

Note:
- $[\boldsymbol{r}]_{\text{pci}}$ and $\left[^{\text{pci}}\boldsymbol{v} \right]_{\text{pci}}$ can be input in any units, but they *must* be consistent.

Algorithm 64: `rot_pci2rsw`

Rotation matrix from Earth-centered inertial (PCI) frame to Hill/orbital (RSW) frame.

Inputs:
- $[r]_{\text{pci}} \in \mathbb{R}^3$ - position expressed in PCI frame
- $\left[^{\text{pci}}v\right]_{\text{pci}} \in \mathbb{R}^3$ - PCI velocity expressed in PCI frame

Procedure:

$$\mathbf{R}_{\text{pci}\rightarrow\text{rsw}} = \texttt{rot\_rsw2pci}\left([r]_{\text{pci}}, \left[^{\text{pci}}v\right]_{\text{pci}}\right)^T \qquad \text{(Algorithm 63)}$$

return $\mathbf{R}_{\text{pci}\rightarrow\text{rsw}}$

Outputs:
- $\mathbf{R}_{\text{pci}\rightarrow\text{rsw}} \in \mathbb{R}^{3\times3}$ - passive rotation matrix from PCI frame to RSW frame

Note:
- $[r]_{\text{pci}}$ and $\left[^{\text{pci}}v\right]_{\text{pci}}$ can be input in any units, but they *must* be consistent.

TODO: get orbital angular velocity TODO: everything below

$$v = \dot{r}\hat{R} + r\dot{\nu}\hat{S} = \dot{\nu}\left(\frac{dr}{d\nu}\hat{R} + r\hat{S}\right) \tag{8.10}$$

$$\omega_{IJK\rightarrow RSW} = \dot{\Omega}\hat{K} + i n + (\dot{\omega} + \dot{\nu})\,\hat{W} \tag{8.11}$$

Angle between T and v is the flight path angle:

$$\therefore v_T = \frac{h}{r} = \frac{\mu(1 + e\cos\nu)}{\sqrt{a\mu(1 - e^2)}}$$

In the radial direction,

$$v_R = \dot{r} = \frac{dr}{dt} = \frac{d}{dt}\left[\frac{h^2}{\mu}\frac{1}{1 + e\cos\mu}\right] = \frac{h^2}{\mu}\frac{d}{dt}\left[\frac{1}{1 + e\cos\mu}\right] = \frac{h^2}{\mu}\left[\frac{e\sin\nu}{(1 + e\cos\nu)^2}\frac{h}{r^2}\right]$$

$$\therefore v_R = \frac{\mu e\sin\nu}{\sqrt{a\mu(1 - e^2)}}$$

9

Measurement of Time

TODO: update all tikz diagrams in separate latex doc

9.1 Time Units

An **epoch** (pronounced "epic") is a moment in time that designates some event, or more broadly designates some reference point in time. We refer to a particular instant in time as a **date**. Thus, an epoch is described by a date. Two commonly used epochs are shown below[1]:

$$\boxed{\text{J2000.0 epoch} = 2000 \text{ January 1, 12:00:00.000 TT}} \tag{9.1}$$

$$\boxed{\text{GPS epoch} = 1980 \text{ January 6, 00:00:00.000 UTC/GPS}} \tag{9.2}$$

To be able to determine the epoch of an event, we need both a precise understanding of time and systems of units that are universally agreed upon [103, pp. 174, 183].

9.1.1 Gregorian (Calendar) Dates

The classical way to define an epoch is using a **Gregorian date** in the format YYYY/MM/DD hh:mm:ss. YYYY is the year, MM the month, DD the day, hh the hour, mm the minute, and ss the second. Note that ss can be fractional (i.e. we could have ss = 5.12904...). From a computational perspective, it is convenient to store the Gregorian date in a row vector, $\mathbf{c}$:

$$\boxed{\mathbf{c} = \begin{bmatrix} \text{YYYY} & \text{MM} & \text{DD} & \text{hh} & \text{mm} & \text{ss} \end{bmatrix} \quad \text{(Gregorian date)}} \tag{9.3}$$

The symbols in the vector above are meant to be used as variables in the various unit conversions. The shorthand for the actual units (including centuries) are defined as follows:
- c – centuries
- y – years
- mo – months
- wk – weeks

---

[1] The TT, UTC, and GPS time scales are discussed in Section 9.2.

- d – days
- h – hours
- m – minutes
- s – seconds

An example of a date written as a Gregorian date is shown below.

$$\text{February 1, 2022 16:07:13}$$

Oftentimes, numbers and variables are also subscripted with the units:

$$\text{February 1, 2022 } 16^{\text{h}}07^{\text{m}}13^{\text{s}}$$

Finally, in everyday usage, the Gregorian date is often referred to simply as the **calendar date**. Note that this is a somewhat of a misnomer; since there are different calendars, such as the Julian and Gregorian calendars, "calendar date" can be somewhat ambiguous. However, most of the world now uses the Gregorian calendar, and we use the abbreviation "`cal`" to define the name of functions as well as variables within those functions.

Gregorian (calendar) date $\leftrightarrow$ day of year

In some applications, it is necessary to find the day of the year from the calendar date, and vice-versa. Algorithm 65 for finding the day of year from the Gregorian (calendar) date was developed from the discussion in [103, p. 200].

Algorithm 65: `cal2doy`
Day of year from Gregorian (calendar) date.

Inputs:
- $\mathbf{c} \in \mathbb{R}^{1 \times 3}$ - Gregorian (calendar) date $[\text{y}, \text{mo}, \text{d}]$

Procedure:
1. Extract YYYY [y], MM [mo], and DD[d] from $\mathbf{c}$.
2. Vector to store number of days in each month in non-leap years [d].

$$\mathbf{d} = \begin{bmatrix} 31 & 28 & 31 & 30 & 31 & 30 & 31 & 31 & 30 & 31 & 30 & 31 \end{bmatrix}$$

3. Change the number of days in February to 29 if in a leap year [d].

$$\textbf{if } (\text{mod}\,(\text{YYYY}, 4) = 0)$$
$$\qquad d_2 = 29$$
$$\textbf{end}$$

4. Day of year [d].

$$\text{DOY} = \text{DD} + \sum_{i=1}^{\text{MM}-1} d_i$$

5. Return the result.

$$\textbf{return } \text{DOY}$$

Outputs:
- $\text{DOY} \in \mathbb{Z}$ - day of year [d]

Test Cases:
- See Appendix A.2.1.

To perform the conversion in the opposite direction, we can use Algorithm 66. Note that for the inverse conversion, we require the year as an input, since we need to know if it is a leap year or not.

Algorithm 66: `doy2cal`
Day of year from Gregorian (calendar) date.

Inputs:
- $\text{YYYY} \in \mathbb{Z}$　- year $[\text{y}]$
- $\text{DOY} \in \mathbb{Z}$　- day of year $[\text{d}]$

Procedure:

1. Vector to store number of days in each month in non-leap years $[\text{d}]$.

$$\mathbf{d} = \begin{bmatrix} 31 & 28 & 31 & 30 & 31 & 30 & 31 & 31 & 30 & 31 & 30 & 31 \end{bmatrix}$$

2. Change the number of days in February to 29 if in a leap year $[\text{d}]$.

if $(\mathrm{mod}\,(\text{YYYY}, 4) = 0)$
$\quad \big| \quad d_2 = 29$
end

3. Determine the month $[\text{mo}]$.

$\text{MM} = 1$
while $\left(\sum_{i=1}^{\text{MM}} d_i < \text{DOY} \right)$
$\quad \big| \quad \text{MM} = \text{MM} + 1$
end

4. Determine the day of the month $[\text{d}]$.

if $(\text{MM} = 1)$
$\quad \big| \quad \text{DD} = \text{DOY}$
else
$\quad \big| \quad \text{DD} = \text{DOY} - \sum_{i=1}^{\text{MM}-1} d_i$
end

5. Define the $\mathbf{c}$ vector storing the Gregorian (calendar) date.

$$\mathbf{c} = \begin{bmatrix} \text{YYYY} & \text{MM} & \text{DD} & 0 & 0 & 0 \end{bmatrix}$$

6. Return the result.

return $\mathbf{c}$

Outputs:
- $\mathbf{c} \in \mathbb{R}^{1 \times 6}$ - Gregorian (calendar) date $[\mathrm{y}, \mathrm{mo}, \mathrm{d}, \mathrm{h}, \mathrm{m}, \mathrm{s}]$

Note:
- This implementation is only valid between the years 1901 and 2099.

Test Cases:
- See Appendix A.2.1.

9.1.2   Julian and Modified Julian Dates

Julian date

The **Julian date**, JD, is the number of days from the epoch January 1, 4713 B.C., 12:00 [60, p. 319].

Modified Julian date

The **modified Julian date**, MJD, is defined as [60, p. 319]

$$\boxed{\mathrm{MJD} = \mathrm{JD} - 2400000.5} \tag{9.4}$$

If we know Julian date, then we can use Algorithm 68 to calculate the modified Julian date.

Algorithm 67: `jd2mjd`
Modified Julian date from Julian date.

Inputs:
- $\mathrm{JD} \in \mathbb{R}$ - Julian date $[\mathrm{d}]$

Procedure:

   $\mathrm{MJD} = \mathrm{JD} - 2400000.5$
   return MJD

Outputs:
- $\mathrm{MJD} \in \mathbb{R}$ - modified Julian date $[\mathrm{d}]$

Test Cases:
- See Appendix A.2.1.

Rearranging Eq. (9.4),

$$\boxed{\mathrm{JD} = \mathrm{MJD} + 2400000.5} \tag{9.5}$$

We formalize this as Algorithm 68.

Algorithm 68: `mjd2jd`
Julian date from modified Julian date.

Inputs:
- $\mathrm{MJD} \in \mathbb{R}$ - modified Julian date [d]

Procedure:

$$\mathrm{JD} = \mathrm{MJD} + 2400000.5$$
return JD

Outputs:
- $\mathrm{JD} \in \mathbb{R}$ - Julian date [d]

Test Cases:
- See Appendix A.2.1.

Julian centuries since J2000.0

The variable T is used to denote **Julian centuries since J2000.0**. It is defined as

$$T = \frac{\mathrm{JD} - 2451545}{36525} \qquad (9.6)$$

Note that T and JD are in the same time scale (see Section 9.2 for more information on time scales). For example, if we are given the Julian date of TT, $\mathrm{JD}_{\mathrm{TT}}$, then Eq. (9.6) will give us T_{TT} [103, p. 184]. Formalizing this equation as Algorithm 69,

Algorithm 69: `jd2t`
Julian centuries since J2000.0 from Julian date.

Inputs:
- $\mathrm{JD} \in \mathbb{R}$ - Julian date [d]

Procedure:

$$T = \frac{\mathrm{JD} - 2451545}{36525}$$
return T

Outputs:
- $T \in \mathbb{R}$ - Julian centuries since J2000.0 [c]

Test Cases:
- See Appendix A.2.1.

Note that if we substitute Eq. (9.5) into Eq. (9.6), we can also write T in terms of MJD as

$$T = \frac{(\text{MJD} + 2400000.5) - 2451545}{36525} \quad \rightarrow \quad \boxed{T = \frac{\text{MJD} - 51544.5}{36525}} \tag{9.7}$$

Formalizing Eq. (9.7) as Algorithm 70,

Algorithm 70: `mjd2t`
Julian centuries since J2000.0 from Julian date.

Inputs:

 • $\text{MJD} \in \mathbb{R}$ - modified Julian date [d]

Procedure:

 $T = \dfrac{\text{MJD} - 51544.5}{36525}$

 return T

Outputs:

 • $T \in \mathbb{R}$ - Julian centuries since J2000.0 [c]

Test Cases:

 • See Appendix A.2.1.

Finally, it is *essential* to note that T_{GPS}, T_{TAI}, T_{UT1}, T_{UTC} do ***not*** represent **exact** centuries since the J2000.0 epoch. This is because the J2000.0 epoch is defined as 2000 January 1, 12:00:00.000 TT, which corresponds to a modified Julian date of TT of $(\text{MJD}_{\text{TT}})_{\text{J2000.0}} = 51544.5$. However, at the J2000.0 epoch,

$$(\text{MJD}_{\text{GPS}})_{\text{J2000.0}} \neq (\text{MJD}_{\text{TAI}})_{\text{J2000.0}} \neq (\text{MJD}_{\text{UT1}})_{\text{J2000.0}} \neq (\text{MJD}_{\text{UTC}})_{\text{J2000.0}} \neq 51544.5$$

Nonetheless, the Julian centuries since J2000.0 for *any* of the time scales is defined using Eqs. (9.6) and (9.7); any equations/algorithm that make use of T are already written accordingly [68, pp. 45, 52], [103, p. 184].

Gregorian (calendar) date $\leftrightarrow$ modified Julian date

Algorithm 71 below (adapted from Appendix A.1.1 in [60, pp. 321–322] and) converts a Gregorian (calendar) date to a modified Julian date. Note that the aforementioned references account for dates before 1582 October 15[2]. However, to simplify our function, we assume the date is on or after October 15, 1582, which is true for virtually all practical applications today.

---

[2] Due to the Gregorian calendar reform, there are no dates between 1582 October 4 and 1582 October 15 [38], [61, p. 15]; the algorithms presented [61, pp. 14–15] and [60, pp. 321–322] account for this. Note that there appears to be a slight error Eq. (A.5) in [60, p. 321]; it uses October 10, 1582, which is a date that does not exist.

Algorithm 71: `cal2mjd`
Modified Julian date from Gregorian (calendar) date.

Inputs:
- $\mathbf{c} \in \mathbb{R}^{1 \times 6}$ - Gregorian (calendar) date $[y, mo, d, h, m, s]$

Procedure:
1. Extract YYYY, MM, DD, hh, mm, and ss from $\mathbf{c}$.
2. Throw an error if the date is before 1582 October 15.

> **if** $(\text{YYYY} \leq 1582)$ **and** $[(\text{MM} < 10)$ **or** $((\text{MM} = 10)$ **and** $(\text{DD} < 15))]$
> > error("This function is only valid for dates after 1582 October 14.")
>
> **end**

3. Convert time to fraction of a day (Algorithm 73) $[d]$.

$$f_{\text{DD}} = \texttt{hms2f}(\text{hh}, \text{mm}, \text{ss})$$

4. Append time to day $[d]$.

$$\text{DD} = \text{DD} + f_{\text{DD}}$$

5. Handle leap years.

> **if** $\text{MM} \leq 2$
> > $y = \text{YYYY} - 1$
> > $m = \text{MM} + 12$
>
> **else**
> > $y = \text{YYYY}$
> > $m = \text{MM}$
>
> **end**

6. Handle leap days.

$$B = \left\lfloor \frac{y}{400} \right\rfloor - \left\lfloor \frac{y}{100} \right\rfloor + \left\lfloor \frac{y}{4} \right\rfloor$$

7. Modified Julian date.

$$\text{MJD} = 365y - 679004 + B + \lfloor 30.6001(m+1) \rfloor + \text{DD}$$

8. Return the result.

> **return** MJD

Outputs:
- $\text{MJD} \in \mathbb{R}$ - modified Julian date $[d]$

Note:
- This implementation is only valid for dates after 1582 October 14.

Test Cases:
- See Appendix A.2.1.

To perform the conversion in the opposite direction, we can use Algorithm 72, which is adapted from Appendix A.1.2 in [60, pp. 322–323]. Similar to how Algorithm 71 does not support dates before 1582 October 15, Algorithm 72 does not support modified Julian dates before -100840 (corresponding to 1582 October 15). Note that the procedure is also slightly modified to return the fraction of the day in hours, minutes, and seconds as well.

Algorithm 72: `mjd2cal`
Gregorian (calendar) date from modified Julian date.

Inputs:
- $\mathrm{MJD} \in \mathbb{R}$ - modified Julian date $[\mathrm{d}]$

Procedure:

1. Integer Julian Day (Julian date at noon) $[\mathrm{JD}]$.

$$a = \lfloor \mathrm{MJD} \rfloor + 2400001$$

2. Auxiliary quantities.

$$b = \left\lfloor \frac{a - 1867216.25}{36524.25} \right\rfloor$$

$$c = a + b - \left\lfloor \frac{b}{4} \right\rfloor + 1525$$

$$d = \left\lfloor \frac{c - 121.1}{365.25} \right\rfloor$$

$$e = \lfloor 365.25d \rfloor$$

$$f = \left\lfloor \frac{c - e}{30.6001} \right\rfloor$$

3. Day of month $[\mathrm{d}]$.

$$\mathrm{DD} = c - e - \lfloor 30.6001f \rfloor + f_{\mathrm{DD}}$$

4. Month of year $[\mathrm{mo}]$.

$$\mathrm{MM} = f - 1 - 12 \left\lfloor \frac{f}{14} \right\rfloor$$

5. Year $[\mathrm{y}]$.

$$\mathrm{YYYY} = d - 4715 - \left\lfloor \frac{7 + \mathrm{MM}}{10} \right\rfloor$$

6. Fraction of day (Algorithm 75) $[\mathrm{d}]$.

$$f_{\mathrm{DD}} = \texttt{mjd2f}(\mathrm{MJD})$$

7. Hours, minutes, and seconds from fraction of day (Algorithm 74) $[\mathrm{h}, \mathrm{m}, \mathrm{s}]$.

$$[\mathrm{hh}, \mathrm{mm}, \mathrm{ss}] = \texttt{f2hms}(f_{\mathrm{DD}})$$

8. Define the **c** vector storing the Gregorian (calendar) date $[\mathrm{y}, \mathrm{mo}, \mathrm{d}, \mathrm{h}, \mathrm{m}, \mathrm{s}]$.

$$\mathbf{c} = \begin{bmatrix} \mathrm{YYYY} & \mathrm{MM} & \mathrm{DD} & \mathrm{hh} & \mathrm{mm} & \mathrm{ss} \end{bmatrix}$$

9. Return the result.

       return c

Outputs:
- $\mathbf{c} \in \mathbb{R}^{1 \times 6}$ - Gregorian (calendar) date $[y, mo, d, h, m, s]$

Note:
- This implementation is only valid for dates after 1582 October 14.

Test Cases:
- See Appendix A.2.1.

9.1.3 Time Measurement Within a Day

The time within a single day is typically tracked with hours, minutes, and seconds.

Fraction of a day $\leftrightarrow$ hours, minutes, seconds

Let f_{DD} represent the fraction of a day. To obtain the fraction of the day from the elapsed hours (hh), minutes (mm), and seconds (ss) since the beginning of the day,

$$f_{\mathrm{DD}} = \left[\frac{\mathrm{hh}}{24}\right] + \left[\frac{\mathrm{mm}}{(24 \text{ h/d})(60 \text{ m/h})}\right] + \left[\frac{\mathrm{ss}}{(24 \text{ h/d})(60 \text{ m/h})(60 \text{ s/m})}\right]$$

$$\boxed{f_{\mathrm{DD}} = \left(\frac{\mathrm{hh}}{24}\right) + \left(\frac{\mathrm{mm}}{1440}\right) + \left(\frac{\mathrm{ss}}{86400}\right)} \tag{9.8}$$

We formalize this as Algorithm 73.

Algorithm 73: `hms2f`
Fraction of day from hours, minutes, and seconds.

Inputs:
- $\mathrm{hh} \in \mathbb{Z}$ - hours $[\mathrm{h}]$
- $\mathrm{mm} \in \mathbb{Z}$ - minutes $[\mathrm{m}]$
- $\mathrm{ss} \in \mathbb{R}$ - seconds $[\mathrm{s}]$

Procedure:

$$f_{\mathrm{DD}} = \left(\frac{\mathrm{hh}}{24}\right) + \left(\frac{\mathrm{mm}}{1440}\right) + \left(\frac{\mathrm{ss}}{86400}\right)$$

   return f_{DD}

Outputs:
- $f_{\mathrm{DD}} \in \mathbb{R}$ - fraction of day $[\mathrm{d}]$

Test Cases:
- See Appendix A.2.1.

To perform the conversion in the opposite direction, we can use Algorithm 74 [103, p. 199].

Algorithm 74: f2hms
Hours, minutes, and seconds from fraction of day.

Inputs:
- $f_{DD} \in \mathbb{R}$ - fraction of day [d]

Procedure:
1. Hours.

$$\text{hh} = \lfloor 24 f_{DD} \rfloor$$

2. Minutes.

$$\text{mm} = \lfloor 60(24 f_{DD} - \text{hh}) \rfloor$$

3. Seconds.

$$\text{ss} = 3600 \left(24 f_{DD} - \text{hh} - \frac{\text{mm}}{60} \right)$$

4. Return the result.

$$\textbf{return } \text{hh, mm, ss}$$

Outputs:
- $\text{hh} \in \mathbb{Z}$ - hours [h]
- $\text{mm} \in \mathbb{Z}$ - minutes [m]
- $\text{ss} \in \mathbb{R}$ - seconds [s]

Test Cases:
- See Appendix A.2.1.

Modified Julian date → fraction of a day

Given a positive modified Julian date, $\text{MJD} > 0$, we can easily determine the fraction of the day as

$$f_{DD} = \text{MJD} - \lfloor \text{MJD} \rfloor, \quad \text{MJD} > 0$$

Note that this also works for negative modified Julian dates. We formalize this as Algorithm 75.

Algorithm 75: mjd2f
Fraction of day from modified Julian date.

Inputs:
- $\text{MJD} \in \mathbb{R}$ - modified Julian date [d]

Procedure:

$$f_{DD} = \text{MJD} - \lfloor \text{MJD} \rfloor$$
$$\textbf{return } f_{DD}$$

Outputs:
  • $f_{DD} \in \mathbb{R}$ - fraction of day [d]

Test Cases:
  • See Appendix A.2.1.

9.2   Time Scales

Measuring Time

There are five main ways for measuring time that we consider for astrodynamic purposes:

1. **Solar Time:** Defined by successive transits of the Sun over a particular meridian (i.e. line of longitude). Specifically, one solar day is the time required for an observer on the Earth to revolve once *and* observe the Sun in the same location [103, p. 177].

2. **Sidereal Time:** Defined by successive transits of the stars over a particular meridian (i.e. line of longitude) [103, p. 176]. Sidereal time is similar to solar time, but it "baselines" time against the stars and not against the Sun.
   • Since an Earth-fixed frame is effectively motionless with respect to distant stars, the Earth will rotate through exactly $360°$ in a sidereal day.
   • In contrast, due to the orbital motion of the Earth around the Sun, the Earth will move approximately $1°$ in its orbit around the Sun over the course of the day (since there just over 360 days in a year), so the Earth will rotate through approximately $361°$ in a solar day. Thus, solar time and sidereal time increase at different rates.

3. **Universal Time:** Universal time was originally meant to define a standard for solar time by measuring time via observing successive transits of a *fictitious mean Sun* over the Greenwich meridian [102], [103, pp. 177–178] (i.e. universal time = mean solar time at Greenwich [103, p. 178]). However, due to the difficulty in precisely measuring the position of the Sun, the original realization of Universal time (called UT0) used measurements of the positions of stars. Nowadays, we use UT1 and UTC, which are covered further below.

4. **Atomic Time:** Both solar and sidereal time do not increase at a constant rate due to many factors (the Earth wobbles, the Earth's orbit is slightly elliptical, etc.). Atomic time is baselined with respect to the specific quantum transition of electrons in a cesium-133 atom. Since this transition is extremely deterministic, atomic time essentially represents a way to measure time in a constant fashion [103, p. 190].
   • Note that the length of a second was also chosen to be based loosely on solar time, and not sidereal time.

5. **Dynamical Time:** All the times above are traditionally defined as being measured on the Earth. However, due to our motion with respect to "true" inertial space and the gravity we experience, our measurement of time is fundamentally different to any other point in the universe due to general relativity. This is consequential when generating planetary ephemerides, since our experience of time is different than elsewhere in the solar system. Dynamical time is meant to measure time in such a way that it can be an independent variable of the equations of celestial mechanics. There have been many definitions of dynamical time in the previous decades, many of which were based on measurements of the SI second in different reference frames. However, this would mean that the rate of dynamical time differed from the SI second on Earth. The current realization of dynamical time uses a rate that matches the SI second on Earth [29], [103, pp. 190–193].

Time Scales

There are **five** main time scales we use in most practical applications [103, pp. 179–181, 190]:

1. **UT1 (Universal Time 1):** Universal time that accounts for the polar motion of the Earth. Since the polar motion of the Earth is not uniform, UT1 does not increase in a constant manner [102], [103, p. 179].
2. **UTC (Coordinate Universal Time):** UTC is designed to follow UT1, but increases constantly and is calculated using atomic clocks. Since UT1 varies irregularly due to variations in the Earth's rotation (as mentioned above), leap seconds are used to keep UTC within $\pm 0.9^{\text{s}}$ of UT1 [102], [103, pp. 180–181].
3. **GPS:** GPS time increases constantly at the same rate as UTC time[3], but it does *not* use leap seconds to stay within $\pm 0.9^{\text{s}}$ of UT1. Therefore, it differs from UTC by the number of leap seconds [103, p. 181].
4. **TAI (International Atomic Time):** TAI is defined independent of the average rotation of the Earth. It is based on counting the cycles of a high-frequency electrical circuit maintained in resonance with a cesium-133 atomic transition, and serves as the source of truth for defining a constant, linearly increasing definition of time, which is the **SI second**. Thus, UTC and GPS are based on offsets to TAI. UTC is offset from TAI by a certain number of leap seconds (since UTC is designed to stay within $\pm 0.9^{\text{s}}$ of UT1), and GPS is always offset from TAI by 19 seconds [103, pp. 177, 190].
5. **TT (Terrestrial Time):** TT is the realization of terrestrial dynamical time (i.e. dynamical time for the Earth). Although atomic time (realized by TAI, and used to define GPS and UTC time) is defined to realize atomic time, we actually use the TAI time scale to define dynamical time in the form of TT time[4]. Due to its historical definition, TT time is offset from TAI time by 32.184 seconds [103, pp. 190–191].

One time scale that is important to know about, but is generally *not* used in practice (unless the extra fidelity is needed), is **barycentric dynamical time (TDB)**. It is rigorously defined as the "independent variable of [the] equations of motion with respect to the barycenter of the solar system" [103, p. 191]; thus, it is meant to be used with the planetary ephemerides provided by JPL. However, recall the discussion earlier in this section about how earlier realizations of dynamical time had different rates to "Earth" time. In contrast, TDB is specifically defined so that a second in barycentric dynamical time matches a second as measured on Earth; this allows TDB to follow TT closely, to a maximum difference of 1.7 ms. Thus, in practice we use terrestrial time (TT) instead of TDB – for most models using TDB, the effect of the difference between TT and TDB is largely insignificant [68, p. 61], [103, p. 193].

9.2.1 Converting Between Time Scales

The relationships between the various time scales can be summarized as [103, p. 190]

$$\text{UTC} = \text{UT1} - \Delta\text{UT1}$$
$$\text{TAI} = \text{UTC} + \Delta\text{AT}$$
$$\text{TT} = \text{TAI} + 32.184^{\text{s}}$$
$$\text{TAI} = \text{GPS} + 19.0^{\text{s}}$$

A discussion of how to obtain ΔUT1 and ΔAT is included at the end of this section.

In general, we typically have UTC and UT1 as modified Julian dates, and ΔUT1 in seconds. Thus,

$$\text{MJD}_{\text{UTC}} = \text{MJD}_{\text{UT1}} - \left[\frac{(\Delta\text{UT1})^{\text{s}}}{86400}\right] \qquad \leftrightarrow \qquad \text{MJD}_{\text{UT1}} = \text{MJD}_{\text{UTC}} + \left[\frac{(\Delta\text{UT1})^{\text{s}}}{86400}\right] \qquad (9.9)$$

The conversion factor (1/86400) comes from the fact that there are $(24)(3600) = 86400$ seconds in a day. Similarly, ΔAT is also typically expressed in seconds. Thus, we have

$$\text{MJD}_{\text{TAI}} = \text{MJD}_{\text{UTC}} + \left[\frac{(\Delta\text{AT})^{\text{s}}}{86400}\right] \qquad \leftrightarrow \qquad \text{MJD}_{\text{UTC}} = \text{MJD}_{\text{TAI}} - \left[\frac{(\Delta\text{AT})^{\text{s}}}{86400}\right] \qquad (9.10)$$

---

[3] To within 1 microsecond per day [103, p. 181].
[4] This is because TAI is measured on Earth, so it is essentially an approximation of the true dynamical time of Earth.

The remaining two conversions can be written as

$$\boxed{\mathrm{MJD_{TT}} = \mathrm{MJD_{TAI}} + \left(\frac{32.184}{86400}\right) \qquad \leftrightarrow \qquad \mathrm{MJD_{TAI}} = \mathrm{MJD_{TT}} - \left(\frac{32.184}{86400}\right)} \tag{9.11}$$

$$\boxed{\mathrm{MJD_{TAI}} = \mathrm{MJD_{GPS}} + \left(\frac{19}{86400}\right) \qquad \leftrightarrow \qquad \mathrm{MJD_{GPS}} = \mathrm{MJD_{TAI}} - \left(\frac{19}{86400}\right)} \tag{9.12}$$

The time systems we use in day-to-day life are based on UTC. Therefore, we typically know an epoch in UTC and wish to convert to the other time scales. This can be achieved with the following basic procedure:

1. Convert UTC to UT1 using Eq. (9.9).
2. Convert UTC to TAI using Eq. (9.10).
3. Convert TAI to TT using Eq. (9.11).
4. Convert TAI to GPS using Eq. (9.12).

For convenience, we also include algorithms for these conversions below.

UTC $\leftrightarrow$ UT1

Algorithm 76: `utc2ut1`
UT1 from UTC.

___

Inputs:
  • $\mathrm{MJD_{UTC}} \in \mathbb{R}$ - UTC (Universal Coordinated Time) $[\mathrm{MJD}]$
  • $(\Delta \mathrm{UT1})^{\mathrm{s}} \in \mathbb{R}$ - difference between UT1 and UTC ($\Delta \mathrm{UT1} = \mathrm{UT1} - \mathrm{UTC}$) $[\mathrm{s}]$

Procedure:

$$\mathrm{MJD_{UT1}} = \mathrm{MJD_{UTC}} + \left[\frac{(\Delta \mathrm{UT1})^{\mathrm{s}}}{86400}\right]$$

Outputs:
  • $\mathrm{MJD_{UT1}} \in \mathbb{R}$ - UT1 (Universal Time 1) $[\mathrm{MJD}]$

Test Cases:
  • See Appendix A.2.2.

Algorithm 77: `ut12utc`
UTC from UT1.

___

Inputs:
  • $\mathrm{MJD_{UT1}} \in \mathbb{R}$ - UT1 (Universal Time 1) $[\mathrm{MJD}]$
  • $(\Delta \mathrm{UT1})^{\mathrm{s}} \in \mathbb{R}$ - difference between UT1 and UTC ($\Delta \mathrm{UT1} = \mathrm{UT1} - \mathrm{UTC}$) $[\mathrm{s}]$

Procedure:

$$\mathrm{MJD_{UTC}} = \mathrm{MJD_{UT1}} - \left[\frac{(\Delta \mathrm{UT1})^{\mathrm{s}}}{86400}\right]$$

Outputs:
  • $\mathrm{MJD_{UTC}} \in \mathbb{R}$ - UTC (Universal Coordinated Time) $[\mathrm{MJD}]$

Test Cases:

- See Appendix A.2.2.

UTC ↔ TAI

Algorithm 78: `utc2tai`

TAI from UTC.

Inputs:
- $\mathrm{MJD}_{\mathrm{UTC}} \in \mathbb{R}$ - UTC (Universal Coordinated Time) $[\mathrm{MJD}]$
- $(\Delta\mathrm{AT})^{\mathrm{s}} \in \mathbb{Z}$ - difference between TAI and UTC ($\Delta\mathrm{AT} = \mathrm{TAI} - \mathrm{UTC}$) $[\mathrm{s}]$

Procedure:

$$\mathrm{MJD}_{\mathrm{TAI}} = \mathrm{MJD}_{\mathrm{UTC}} + \left[\frac{(\Delta\mathrm{AT})^{\mathrm{s}}}{86400}\right]$$

Outputs:
- $\mathrm{MJD}_{\mathrm{TAI}} \in \mathbb{R}$ - TAI (International Atomic Time) $[\mathrm{MJD}]$

Test Cases:

- See Appendix A.2.2.

Algorithm 79: `tai2utc`

UTC from TAI.

Inputs:
- $\mathrm{MJD}_{\mathrm{TAI}} \in \mathbb{R}$ - TAI (International Atomic Time) $[\mathrm{MJD}]$
- $(\Delta\mathrm{AT})^{\mathrm{s}} \in \mathbb{Z}$ - difference between TAI and UTC ($\Delta\mathrm{AT} = \mathrm{TAI} - \mathrm{UTC}$) $[\mathrm{s}]$

Procedure:

$$\mathrm{MJD}_{\mathrm{UTC}} = \mathrm{MJD}_{\mathrm{TAI}} - \left[\frac{(\Delta\mathrm{AT})^{\mathrm{s}}}{86400}\right]$$

Outputs:
- $\mathrm{MJD}_{\mathrm{UTC}} \in \mathbb{R}$ - UTC (Universal Coordinated Time) $[\mathrm{MJD}]$

Test Cases:

- See Appendix A.2.2.

TAI ↔ TT

Algorithm 80: `tai2tt`
TT from TAI.

Inputs:

  • $\mathrm{MJD_{TAI}} \in \mathbb{R}$ - TAI (International Atomic Time) $[\mathrm{MJD}]$

Procedure:

$$\mathrm{MJD_{TT}} = \mathrm{MJD_{TAI}} + \left(\frac{32.184}{86400} \right)$$

Outputs:

  • $\mathrm{MJD_{TT}} \in \mathbb{R}$ - TT (Terrestrial Time) $[\mathrm{MJD}]$

Test Cases:

  • See Appendix A.2.2.

Algorithm 81: `tt2tai`
TAI from TT.

Inputs:

  • $\mathrm{MJD_{TT}} \in \mathbb{R}$ - TT (Terrestrial Time) $[\mathrm{MJD}]$

Procedure:

$$\mathrm{MJD_{TAI}} = \mathrm{MJD_{TT}} - \left(\frac{32.184}{86400} \right)$$

Outputs:

  • $\mathrm{MJD_{TAI}} \in \mathbb{R}$ - TAI (International Atomic Time) $[\mathrm{MJD}]$

Test Cases:

  • See Appendix A.2.2.

TAI $\leftrightarrow$ GPS

Algorithm 82: `tai2gps`
GPS time from TAI.

Inputs:

  • $\mathrm{MJD_{TAI}} \in \mathbb{R}$ - TAI (International Atomic Time) $[\mathrm{MJD}]$

Procedure:

$$\mathrm{MJD_{GPS} = MJD_{TAI}} - \left(\frac{19}{86400}\right)$$

Outputs:

- $\mathrm{MJD_{GPS}} \in \mathbb{R}$ - GPS time [MJD]

Test Cases:

- See Appendix A.2.2.

Algorithm 83: `gps2tai`
TAI time from GPS.

Inputs:

- $\mathrm{MJD_{GPS}} \in \mathbb{R}$ - GPS time [MJD]

Procedure:

$$\mathrm{MJD_{TAI} = MJD_{GPS}} + \left(\frac{19}{86400}\right)$$

Outputs:

- $\mathrm{MJD_{TAI}} \in \mathbb{R}$ - TAI (International Atomic Time) [MJD]

Test Cases:

- See Appendix A.2.2.

GPS $\leftrightarrow$ weeks and seconds

Recall from Section 9.1 that the GPS epoch is defined as

$$\text{GPS epoch} = 1980 \text{ January 6, 00:00:00.000 UTC/GPS}$$

At this epoch, the GPS time was equal to UTC (now, they differ by an integer number of leap seconds). The modified Julian date of the GPS epoch is

$$(\mathrm{MJD_{GPS}})_{\mathrm{GPS}} = 44244$$

The integer number of weeks since the GPS epoch is then

$$\mathrm{wk} = \left\lfloor \frac{\mathrm{MJD_{GPS}} - 44244}{7} \right\rfloor \tag{9.13}$$

The remaining fraction of a week is then

$$f_{\mathrm{wk}} = \left(\frac{\mathrm{MJD_{GPS}} - 44244}{7}\right) - \mathrm{wk}$$

The number of seconds in a week is

$$\left(\frac{3600 \text{ s}}{\mathrm{h}}\right)\left(\frac{24 \text{ h}}{\mathrm{d}}\right)\left(\frac{7 \text{ d}}{\mathrm{wk}}\right) = 604800 \text{ s/wk}$$

The remaining fraction of a week in seconds is then

$$s = 604800 f_{\text{wk}} = 604800 \left[\left(\frac{\text{MJD}_{\text{GPS}} - 44244}{7} \right) - \text{wk} \right]$$

$$\boxed{s = 86400 \left(\text{MJD}_{\text{GPS}} - 44244 \right) - 604800(\text{wk})}$$ (9.14)

Algorithm 84: `gps2wks`
GPS week and seconds from GPS time.

Inputs:
  • MJD_{GPS} - GPS time [MJD]

Procedure:
  1. GPS weeks.

  $$\text{wk} = \left\lfloor \frac{\text{MJD}_{\text{GPS}} - 44244}{7} \right\rfloor$$

  2. GPS seconds.

  $$s = \left(\frac{\text{MJD}_{\text{GPS}} - 44244}{4233600} \right) - \left(\frac{\text{wk}}{604800} \right)$$

Outputs:
  • $\text{wk} \in \mathbb{Z}$ - GPS week [wk]
  • $s \in \mathbb{R}$ - GPS seconds [s]

Test Cases:
  • See Appendix A.2.2.

The conversion in the opposite direction can be performed in a single line as [60, pp. 324–325]

$$\boxed{\text{MJD}_{\text{GPS}} = (7)(\text{wk}) + \left(\frac{s}{86400} \right) + 44244}$$ (9.15)

Algorithm 85: `wks2gps`
GPS time from GPS week and seconds.

Inputs:
  • $\text{wk} \in \mathbb{Z}$ - GPS week [wk]
  • $s \in \mathbb{R}$ - GPS seconds [s]

Procedure:

  $$\text{MJD}_{\text{GPS}} = (7)(\text{wk}) + \left(\frac{s}{86400} \right) + 44244$$

  return MJD_{GPS}

Outputs:
  • MJD_{GPS} - GPS time [MJD]

Test Cases:
- See Appendix A.2.2.

9.2.2 Obtaining the Offsets

Both offsets are tabulated as a function of the modified Julian date. We assume this can be the modified Julian date of any time scale.

Offset Between UT1 and UTC

Recall from Section 9.2.1 that UT1 and UTC time are related by the offset ΔUT1. This offset is reported in various IERS Bulletins:

- Bulletin A: https://datacenter.iers.org/data/latestVersion/6_BULLETIN_A_V2013_016.txt
  - reported in seconds
  - rounded to nearest $(10^{-6})^{\mathrm{s}}$
  - one less digit reported than Bulletin B, but includes predicted values
- Bulletin B: https://datacenter.iers.org/data/latestVersion/207_BULLETIN_B207.txts
  - reported in milliseconds
  - rounded to nearest $(10^{-7})^{\mathrm{s}}$
  - most accurate
- Bulletin D: https://datacenter.iers.org/data/latestVersion/17_BULLETIN_D17.txt
  - reported in seconds
  - rounded to nearest $(10^{-1})^{\mathrm{s}}$
  - least accurate

In practice, it is most useful to download the Earth orientation data files published by IERS. We will need the Earth orientation data files anyway, and the data includes the finalized historical ΔUT1 data reported in seconds and rounded to the nearest $(10^{-7})^{\mathrm{s}}$ (i.e. the values reported in Bulletin A), as well as the predicted values from Bulletin B.

- HTML version: https://datacenter.iers.org/data/html/finals2000A.data.html
- CSV version: https://datacenter.iers.org/data/csv/finals2000A.data.csv
- All versions: https://www.iers.org/IERS/EN/DataProducts/EarthOrientationData/eop.html

To see all versions, navigate to the third link and click on the "`version meta-data`" link next to "`finals.data (IAU2000)`" in the "`Standard EOP data files`" section of the "`Rapid data and predictions`" table. However, it is usually easiest to just use the `.csv` file provided by the second link.

Algorithm 86: `get_dut1`

Obtains the difference between UT1 and UTC ($\Delta\mathrm{UT1} = \mathrm{UT1} - \mathrm{UTC}$).

Inputs:
- $\mathrm{MJD} \in \mathbb{R}$ - modified Julian date of any time scale [MJD]

Procedure:
1. Load the $\Delta\mathrm{UT1}$ vs. MJD data from one of the IERS Earth orientation parameters file discussed previously in this section. Store the $\Delta\mathrm{UT1}$ values in the vector $\mathbf{y} = (y_1, ..., y_N)^T$ and the MJD data in the vector $\mathbf{x} = (y_1, ..., y_N)^T$.
2. Find the number of data points, N, given that $\mathbf{x} \in \mathbb{R}^n$.

$$N = \texttt{length}(\mathbf{x})$$

3. Edge case: the specified date is after the last available date in the data set.

> **if** $\mathrm{MJD} \geq x_N$
> > $\Delta\mathrm{UT1} = y_u$

4. Base case: the specified date is either (a) before the first available date in the data set or (b) contained within the available dates in the data set.

> **else**
> > (a) Find the lower bound of the interval of $\mathbf{x}$ that contains MJD. This can be done using the `find_interval` algorithm from [45]. Note that this algorithm will automatically return the first interval of $\mathbf{x}$ if $\mathrm{MJD} < x_1$.
> >
> > $$[l, \sim] = \texttt{find\_interval}(\mathbf{x}, \mathrm{MJD})$$
> >
> > (b) Extract the leap seconds from the data table.
> >
> > $$\Delta\mathrm{UT1} = y_l$$
> **end**

5. Return the result.

> **return** $\Delta\mathrm{UT1}$

Outputs:
- $\Delta\mathrm{UT1} \in \mathbb{R}$ - difference between UT1 (Universal Time 1) and UTC (Universal Coordinated Time) [s]

Note:
- This algorithm, as well as the `find_interval` algorithm from [45], both assume 1-based indexing, and therefore need to be modified for use in programming languages that use 0-based indexing.

Test Cases:
- See Appendix A.2.2.

Leap Seconds

The offset ΔAT represents the accumulated leap seconds used by UTC to stay within $\pm 0.9^s$ of UT1 (this is what causes the difference between TAI and UTC). It is published in IERS Bulletin C. However, leap seconds are rarely added, so it is easiest to manually define the data from Table 9.1 below in your code.

Table 9.1: Leap second history (ΔAT = TAI − UTC) [41], [50], [103, p. 191].

IERS Bulletin C No.	Date	Modified Julian Date	ΔAT [s]
-	1972 January 1	41317	10
-	1972 July 1	41499	11
-	1973 January 1	41683	12
-	1974 January 1	42048	13
-	1975 January 1	42413	14
-	1976 January 1	42778	15
-	1977 January 1	43144	16
-	1978 January 1	43509	17
-	1979 January 1	43874	18
-	1980 January 1	44239	19
-	1981 July 1	44786	20
-	1982 July 1	45151	21
-	1983 July 1	45516	22
-	1985 July 1	46247	23
-	1988 January 1	47161	24
-	1990 January 1	47892	25
-	1991 January 1	48257	26
-	1992 July 1	48804	27
-	1993 July 1	49169	28
10	1994 July 1	49534	29
10–13	1996 January 1	50083	30
13–16, 16	1997 July 1	50630	31
16–30	1999 January 1	51179	32
30–36	2006 January 1	53736	33
36–43	2009 January 1	54832	34
43–49	2012 July 1	56109	35
49–52	2015 July 1	57204	36
52–63	2017 January 1	57754	37

Algorithm 87: `get_dat`

Obtains the difference between TAI and UTC (i.e. leap seconds) (ΔAT = TAI − UTC).

Inputs:
- $\text{MJD} \in \mathbb{R}$ - modified Julian date of any time scale $[\text{MJD}]$

Procedure:
1. Load the ΔAT vs. MJD data from Table 9.1. Store the ΔAT values in the vector $\mathbf{y} = (y_1, ..., y_N)^T$ and the MJD data in the vector $\mathbf{x} = (y_1, ..., y_N)^T$.
2. Find the number of data points, N, given that $\mathbf{x} \in \mathbb{R}^n$.

$$N = \texttt{length}(\mathbf{x})$$

3. Edge case: the specified date is after the last available date in the data set.

> **if** $\mathrm{MJD} \geq x_N$
> $\quad | \quad \Delta\mathrm{AT} = y_u$

4. Base case: the specified date is either (a) before the first available date in the data set or (b) contained within the available dates in the data set.

> **else**
> (a) Find the lower bound of the interval of **x** that contains MJD. This can be done using the `find_interval` algorithm from [45]. Note that this algorithm will automatically return the first interval of **x** if $\mathrm{MJD} < x_1$.
>
> $$[l, \sim] = \texttt{find\_interval}(\mathbf{x}, \mathrm{MJD})$$
>
> (b) Extract the leap seconds from the data table.
>
> $$\Delta\mathrm{AT} = y_l$$
>
> **end**

5. Return the result.

> **return** $\Delta\mathrm{AT}$

Outputs:
- $\Delta\mathrm{AT} \in \mathbb{R}$ - leap seconds (difference between TAI (International Atomic Time) and UTC (Universal Coordinated Time) [s]

Note:
- This algorithm, as well as the `find_interval` algorithm from [45], both assume 1-based indexing, and therefore need to be modified for use in programming languages that use 0-based indexing.

Test Cases:
- See Appendix A.2.2.

9.3 Angular Units

Our time systems were originally developed based on the rotational motion of the Earth. Therefore, there are multiple angular quantities (see Section 9.4 that are closely related to the notion of time, and are measured using angular units.

Radians ↔ Degrees

Let θ^{rad} represent an angle in radians and θ° represent that same angle in degrees. Then

$$\theta^{\mathrm{rad}} = \left(\frac{\pi}{180}\right)\theta^{\circ} \qquad \leftrightarrow \qquad \theta^{\circ} = \left(\frac{180}{\pi}\right)\theta^{\mathrm{rad}} \tag{9.16}$$

Formalizing Eq. (9.16) as Algorithms 88 and 89,

Algorithm 88: `deg2rad`
Degrees to radians.

Inputs:
- $\theta^\circ \in \mathbb{R}$ - angle in degrees $[^\circ]$

Procedure:

$$\theta^{\mathrm{rad}} = \left(\frac{\pi}{180}\right)\theta^\circ$$

return θ^{rad}

Outputs:
- $\theta^{\mathrm{rad}} \in \mathbb{R}$ - angle in radians $[\mathrm{rad}]$

Test Cases:
- See Appendix A.3.

Algorithm 89: `rad2deg`
Radians to degrees.

Inputs:
- $\theta^{\mathrm{rad}} \in \mathbb{R}$ - angle in radians $[\mathrm{rad}]$

Procedure:

$$\theta^\circ = \left(\frac{180}{\pi}\right)\theta^{\mathrm{rad}}$$

return θ°

Outputs:
- $\theta^\circ \in \mathbb{R}$ - angle in degrees $[^\circ]$

Test Cases:
- See Appendix A.3.

Degrees, Arcminutes, and Arcseconds (Angle Measurement)

Arcminutes and arcseconds are subdivisions of a degree, similar to how minutes and seconds are subdivisions of an hour. Arcminutes are denoted by a single apostrophe, while arcseconds are denoted with two apostrophes [103, p. 175].

$$1' = 1 \text{ arcminute} \qquad 1'' = 1 \text{ arcsecond} \tag{9.17}$$

The relationship between the various units is

$$1^{\circ} = 60' = 3600''$$

Since $1^{\circ} = (\pi/180)$ rad,

$$1^{\circ} = 3600'' = \frac{\pi}{180} \text{ rad} \quad \rightarrow \quad 1 \text{ rad} = \left(\frac{(180)(3600)}{\pi} \right)'' = \left(\frac{648000}{\pi} \right)''$$

Conversely,

$$1'' = \frac{\pi}{648000} \text{ rad}$$

Even smaller units can also be defined by using standard metrix prefixes [56].

$$1 \text{ mas} = 1 \text{ milliarcsecond} = (10^{-3})''$$
$$1 \text{ μas} = 1 \text{ microarcsecond} = (10^{-6})''$$

Typically, we deal with arcseconds. Therefore, we introduce the equations below to convert an angle θ between radians, degrees, and arcseconds.

$$\boxed{\theta'' = (3600)\theta^{\circ} \qquad \leftrightarrow \qquad \theta^{\circ} = \left(\frac{1}{3600} \right) \theta''} \tag{9.18}$$

$$\boxed{\theta'' = \left(\frac{648000}{\pi} \right) \theta^{\text{rad}} \qquad \leftrightarrow \qquad \theta^{\text{rad}} = \left(\frac{\pi}{648000} \right) \theta''} \tag{9.19}$$

Degrees ↔ Arcseconds

Algorithm 90: `deg2arcsec`
Degrees to arcseconds.

Inputs:
- $\theta^{\circ} \in \mathbb{R}$ - angle in degrees $[^{\circ}]$

Procedure:

$$\theta'' = (3600)\theta^{\circ}$$

Outputs:
- $\theta'' \in \mathbb{R}$ - angle in arcseconds $['']$

Test Cases:
- See Appendix A.3.

Algorithm 91: `arcsec2deg`
Arcseconds to degrees.

Inputs:
- $\theta'' \in \mathbb{R}$ - angle in arcseconds $['']$

Procedure:

$$\theta° = \left(\frac{1}{3600} \right) \theta''$$

return $\theta°$

Outputs:
- $\theta° \in \mathbb{R}$ - angle in degrees $[°]$

Test Cases:
- See Appendix A.3.

Radians ↔ Arcseconds

Algorithm 92: `rad2arcsec`
Radians to arcseconds.

Inputs:
- $\theta^{\mathrm{rad}} \in \mathbb{R}$ - angle in radians $[\mathrm{rad}]$

Procedure:

$$\theta'' = \left(\frac{648000}{\pi} \right) \theta^{\mathrm{rad}}$$

return θ''

Outputs:
- $\theta'' \in \mathbb{R}$ - angle in arcseconds $['']$

Test Cases:
- See Appendix A.3.

Algorithm 93: `arcsec2rad`
Arcseconds to radians.

Inputs:
- $\theta'' \in \mathbb{R}$ - angle in arcseconds $['']$

Procedure:

$$\theta^{\text{rad}} = \left(\frac{\pi}{648000} \right) \theta''$$

   return θ^{rad}

Outputs:
   • $\theta^{\text{rad}} \in \mathbb{R}$ - angle in radians [rad]

Test Cases:
   • See Appendix A.3.

Degrees ↔ Degree-Arcminute-Arcsecond

Geographic coordinates are often given in **degree-arcminute-arcsecond (DMS)** format, where the decimal portion of the coordinates (in degrees) are represented using arcminutes and arcseconds. To convert to a pure decimal format, we can use Algorithm 94 below [103, p. 197].

Algorithm 94: `dms2deg`
Degree-minute-second to degrees.

Inputs:
   • $d \in \mathbb{R}$ - degrees [°]
   • $m \in \mathbb{R}$ - arcminutes [′]
   • $s \in \mathbb{R}$ - arcseconds [″]

Procedure:

$$\theta^{\circ} = d + \frac{m}{60} + \frac{s}{3600}$$

   return θ°

Outputs:
   • $\theta^{\circ} \in \mathbb{R}$ - angle in degrees [°]

Test Cases:
   • See Appendix A.3.

Algorithm 95: `deg2dms`
Degrees to degree-minute-second.

Inputs:
   • $\theta^{\circ} \in \mathbb{R}$ - angle in degrees [°]

Procedure:

1. Degree portion $[°]^a$.

$$\mathrm{d} = \mathtt{fix}(\theta°)$$

2. Auxiliary parameter $[°]$.

$$\alpha = \theta° - \mathrm{d}$$

3. Arcminute portion $[']$.

$$\mathrm{m} = \mathtt{fix}(60\alpha)$$

4. Arcsecond portion $['']$.

$$\mathrm{s} = 3600 \left(\alpha - \frac{\mathrm{m}}{60} \right)$$

5. Return the result.

$$\mathbf{return}\ \mathrm{d}, \mathrm{m}, \mathrm{s}$$

Outputs:
- $\mathrm{d} \in \mathbb{R}$ - degrees $[°]$
- $\mathrm{m} \in \mathbb{R}$ - arcminutes $[']$
- $\mathrm{s} \in \mathbb{R}$ - arcseconds $['']$

Test Cases:
- See Appendix A.3.

---

^{a} The $\mathtt{fix}$ operation represents rounding towards zero (i.e. truncation).

Radians $\leftrightarrow$ Degree-Arcminute-Arcsecond

To convert between radians and degree-arcminute-arcsecond form, we can just use the previous two algorithms and convert either the input or output to degrees.

Algorithm 96: `dms2rad`
Degree-minute-second to radians.

Inputs:
- $\mathrm{d} \in \mathbb{R}$ - degrees $[°]$
- $\mathrm{m} \in \mathbb{R}$ - arcminutes $[']$
- $\mathrm{s} \in \mathbb{R}$ - arcseconds $['']$

Procedure:
1. Angle in degrees (Algorithm 94) $[°]$.

$$\theta° = \mathtt{dms2deg}(\mathrm{d}, \mathrm{m}, \mathrm{s})$$

2. Angle in radians (Algorithm 88) $[\mathrm{rad}]$.

$$\theta^{\mathrm{rad}} = \mathtt{deg2rad}(\theta°)$$

3. Return the result.

$$\textbf{return } \theta^{\text{rad}}$$

Outputs:
- $\theta^{\text{rad}} \in \mathbb{R}$ - angle in radians $[\text{rad}]$

Test Cases:
- See Appendix A.3.

Algorithm 97: `rad2dms`
Radians to degree-minute-second.

Inputs:
- $\theta^{\text{rad}} \in \mathbb{R}$ - angle in radians $[\text{rad}]$

Procedure:
1. Angle in degrees (Algorithm 89) $[°]$.

$$\theta^{°} = \texttt{rad2deg}(\theta^{\text{rad}})$$

2. Angle in DMS form (Algorithm 95) $[°,',']$.

$$[\text{d}, \text{m}, \text{s}] = \texttt{deg2dms}(\theta^{°})$$

3. Return the result.

$$\textbf{return } \text{d}, \text{m}, \text{s}$$

Outputs:
- $\text{d} \in \mathbb{R}$ - degrees $[°]$
- $\text{m} \in \mathbb{R}$ - arcminutes $[']$
- $\text{s} \in \mathbb{R}$ - arcseconds $['']$

Test Cases:
- See Appendix A.3.

9.4 Sidereal Time

SI and sidereal seconds

$$t_{\text{UT1}} = t_{\text{UTC}} + \Delta \text{UT1}$$

where ΔUT1 can be found in the IERS Bulletin B.

$$1 \text{ sidereal day} = 86164.0905 \text{ s} = 23 \text{ h } 56 \text{ min } 4.0905 \text{ s} = 23.9344696 \text{ h}$$

$$1 \text{ sidereal s} = 86164.0905 \text{ s} = 23 \text{ h } 56 \text{ min } 4.0905 \text{ s} = 23.9344696 \text{ h}$$

see p. 180 of vallado

9.5 **The Time Object**

TimeConverter object? Time object?

10

Coordinate Frames and Geodesy

10.1 Orbital State Vector and the Earth-Centered Inertial (ECI) Frame

TODO: https://en.wikipedia.org/wiki/Earth-centered_inertial TODO: J2000, GCRS/GCRF, etc.
TODO: update from Section 8.1

The **Earth-Centered Inertial (ECI) coordinate frame** is a coordinate frame fixed with respect to inertial space. Simply put, it remains fixed with respect to the stars. It is defined using the following three basis vectors:

1. $\hat{\boldsymbol{I}}$ – (1st axis) In the direction of the vernal equinox (i.e. in the direction of the **first point of Aries**).
2. $\hat{\boldsymbol{J}}$ – (2nd axis) Normal to $\hat{\boldsymbol{I}}$ and also in the equatorial plane.
3. $\hat{\boldsymbol{K}}$ – (3rd axis) Completes the right-handed triad.

10.2 Earth-Centered Earth-Fixed (ECEF) Frame

10.3 Generic ECEF/ECI Transformations

10.3.1 ECI to ECEF

Consider the position of the satellite resolved in the ECI frame, $[\mathbf{r}]_{\text{eci}}$. We can immediately resolve this vector in the ECEF frame using

$$\boxed{[\mathbf{r}]_{\text{ecef}} = \mathbf{R}_{\text{eci}\to\text{ecef}}[\mathbf{r}]_{\text{eci}}} \tag{10.1}$$

Next, we want the velocity of the satellite relative to the ECEF frame, $^{\text{ecef}}\mathbf{v}$ (i.e. the **ECEF velocity**). $\boldsymbol{\omega}_\oplus$ be the angular velocity of the Earth relative to the ECI frame, and let $\mathbf{v}$ be the **inertial velocity** of the satellite (i.e. its velocity relative to the ECI frame). Since we are going *from* the inertial frame (ECI) *to* the noninertial frame (ECEF),

$$^{\text{ecef}}\mathbf{v} = \mathbf{v} - \boldsymbol{\omega}_\oplus \times \mathbf{r} \tag{10.2}$$

Our goal is to find the ECEF velocity resolved in the ECEF frame, $\left[^{\text{ecef}}\mathbf{v}\right]_{\text{ecef}}$. However, we will have $\mathbf{v}$, $\boldsymbol{\omega}_\oplus$, and $\mathbf{r}$ resolved in the ECI frame. Therefore,

$$\boxed{\left[^{\text{ecef}}\mathbf{v}\right]_{\text{ecef}} = \mathbf{R}_{\text{eci}\to\text{ecef}} \left([\mathbf{v}]_{\text{eci}} - [\boldsymbol{\omega}_\oplus]_{\text{eci}} \times [\mathbf{r}]_{\text{eci}}\right)} \tag{10.3}$$

We formalize the above two calculations as Algorithm 98.

Algorithm 98: `eci2ecef`
ECEF position and velocity from ECI position and velocity.

Inputs:
- $[\mathbf{r}]_{\mathrm{eci}} \in \mathbb{R}^3$ - position resolved in ECI frame
- $[\mathbf{v}]_{\mathrm{eci}} \in \mathbb{R}^3$ - inertial velocity resolved in ECI frame
- $[\boldsymbol{\omega}]_{\oplus} \in \mathbb{R}^3$ - Earth angular velocity resolved in ECI frame $[\mathrm{rad/s}]$
- $\mathbf{R}_{\mathrm{eci}\to\mathrm{ecef}} \in \mathbb{R}^{3\times3}$ - rotation matrix from ECI frame to ECEF frame

Note:
- $[\mathbf{r}]_{\mathrm{eci}}$ and $[\mathbf{v}]_{\mathrm{eci}}$ can be input in any units, but they MUST be consistent. $[\mathbf{r}]_{\mathrm{ecef}}$ and $[^{\mathrm{ecef}}\mathbf{v}]_{\mathrm{ecef}}$ will be output in the same units.
- By definition, the angular velocity of the Earth is defined as the angular velocity of the ECEF frame with respect to the ECI frame.

Procedure:
1. Position resolved in ECEF frame.

$$[\mathbf{r}]_{\mathrm{ecef}} = \mathbf{R}_{\mathrm{eci}\to\mathrm{ecef}}[\mathbf{r}]_{\mathrm{eci}}$$

2. ECEF velocity resolved in ECEF frame.

$$[^{\mathrm{ecef}}\mathbf{v}]_{\mathrm{ecef}} = \mathbf{R}_{\mathrm{eci}\to\mathrm{ecef}}\left([\mathbf{v}]_{\mathrm{eci}} - [\boldsymbol{\omega}_{\oplus}]_{\mathrm{eci}} \times [\mathbf{r}]_{\mathrm{eci}}\right)$$

Outputs:
- $[\mathbf{r}]_{\mathrm{ecef}} \in \mathbb{R}^3$ - position resolved in ECEF frame
- $[^{\mathrm{ecef}}\mathbf{v}]_{\mathrm{ecef}} \in \mathbb{R}^3$ - ECEF velocity resolved in ECEF frame

10.3.2   ECEF to ECI

Consider the position of the satellite resolved in the ECEF frame, $[\mathbf{r}]_{\mathrm{ecef}}$. We can immediately resolve this vector in the ECI frame using

$$\boxed{[\mathbf{r}]_{\mathrm{eci}} = \mathbf{R}_{\mathrm{eci}\to\mathrm{ecef}}[\mathbf{r}]_{\mathrm{ecef}}} \tag{10.4}$$

Next, we want the inertial velocity of the satellite, $\mathbf{v}$ (i.e. the **ECI velocity**, or the velocity relative to the ECI frame). Rearranging Eq. (**??**), we find

$$\mathbf{v} = {}^{\mathrm{ecef}}\mathbf{v} + \boldsymbol{\omega}_{\oplus} \times \mathbf{r} \tag{10.5}$$

We typically have $\boldsymbol{\omega}_{\oplus}$ already resolved in the ECI frame, and $^{\mathrm{ecef}}\mathbf{v}$ resolved in the ECEF frame. Additionally, from Eq. (10.4), we already have $\mathbf{r}$ resolved in the ECI frame. Thus, we write

$$\boxed{[\mathbf{v}]_{\mathrm{eci}} = \mathbf{R}_{\mathrm{ecef}\to\mathrm{eci}}[^{\mathrm{ecef}}\mathbf{v}]_{\mathrm{ecef}} + [\boldsymbol{\omega}_{\oplus}]_{\mathrm{eci}} \times [\mathbf{r}]_{\mathrm{eci}}} \tag{10.6}$$

We formalize the above two calculations as Algorithm 99.

Algorithm 99: `ecef2eci`
ECI position and velocity from ECEF position and velocity.

Inputs:

- $[\mathbf{r}]_{\text{ecef}} \in \mathbb{R}^3$ - position resolved in ECEF frame
- $[^{\text{ecef}}\mathbf{v}]_{\text{ecef}} \in \mathbb{R}^3$ - ECEF velocity resolved in ECEF frame
- $[\boldsymbol{\omega}]_{\oplus} \in \mathbb{R}^3$ - Earth angular velocity resolved in ECI frame $[\text{rad/s}]$
- $\mathbf{R}_{\text{ecef}\to\text{eci}} \in \mathbb{R}^{3\times3}$ - rotation matrix from ECEF frame to ECI frame

Note:
- $[\mathbf{r}]_{\text{ecef}}$ and $[^{\text{ecef}}\mathbf{v}]_{\text{ecef}}$ can be input in any units, but they MUST be consistent. $[\mathbf{r}]_{\text{eci}}$ and $[\mathbf{v}]_{\text{eci}}$ will be output in the same units.
- By definition, the angular velocity of the Earth is defined as the angular velocity of the ECEF frame with respect to the ECI frame.

Procedure:
1. Position resolved in ECI frame.

$$[\mathbf{r}]_{\text{eci}} = \mathbf{R}_{\text{ecef}\to\text{eci}}[\mathbf{r}]_{\text{ecef}}$$

2. Inertial velocity resolved in ECI frame.

$$[\mathbf{v}]_{\text{eci}} = \mathbf{R}_{\text{ecef}\to\text{eci}}[^{\text{ecef}}\mathbf{v}]_{\text{ecef}} + [\boldsymbol{\omega}_{\oplus}]_{\text{eci}} \times [\mathbf{r}]_{\text{eci}}$$

Outputs:
- $[\mathbf{r}]_{\text{eci}} \in \mathbb{R}^3$ - position resolved in ECI frame
- $[\mathbf{v}]_{\text{eci}} \in \mathbb{R}^3$ - inertial velocity resolved in ECI frame

10.4   Simple Transformations for Pure Rotation

For the case of pure rotation of the Earth about its 3rd axis, we assume the following:

$$[\boldsymbol{\omega}_{\oplus}]_{\text{eci}} = \begin{bmatrix} 0 \\ 0 \\ \omega_{\oplus} \end{bmatrix} \tag{10.7}$$

$$\mathbf{R}_{\text{eci}\to\text{ecef}} = \mathbf{R}_3(\theta_{\text{GMST}}) \tag{10.8}$$

For the transformation in the opposite direction, we have

$$\mathbf{R}_{\text{ecef}\to\text{eci}} = \mathbf{R}_{\text{eci}\to\text{ecef}}^T = \mathbf{R}_3(\theta_{\text{GMST}})^T = \mathbf{R}_3(-\theta_{\text{GMST}}) \tag{10.9}$$

where $\omega_{\oplus}$ is the rotation rate of the Earth (see Appendix **??**) and θ_{GMST} is the Greenwich mean sidereal time (see Algorithms **??** and **??** in Section **??**) [**damico_279a_lecture_6**]. We implement these equations as Algorithms **??**–102.

Algorithm 100: `w_earth_approx`
Angular velocity of the Earth resolved in the ECI frame (approximate).

Procedure:
1. Define $\omega_{\oplus}$ $[\text{rad/s}]$ using the value from Appendix **??**.

2. Angular velocity of the Earth resolved in the ECI frame $[\mathrm{rad/s}]$.

$$[\boldsymbol{\omega}_{\oplus}]_{\mathrm{eci}} = \begin{bmatrix} 0 \\ 0 \\ \omega_{\oplus} \end{bmatrix}$$

return • $[\boldsymbol{\omega}_{\oplus}]_{\mathrm{eci}} \in \mathbb{R}^3$ - Earth angular velocity resolved in ECI frame $[\mathrm{rad/s}]$

Algorithm 101: `eci2ecef_matrix_approx`

Rotation matrix from ECI frame to ECEF frame (approximate).

Inputs:
   • $\mathrm{MJD}_{\mathrm{UT1}} \in \mathbb{R}$ - UT1 (Universal Time 1) $[\mathrm{MJD}]$
   • $\mathtt{simple} \in \mathbb{B}$ - *(OPTIONAL)* `true` if simpler calculation of GMST should be
                                 used, `false` otherwise (defaults to `false`)

Procedure:
   1. Default `simple` to `false` if not input.

> **if** (`simple` not input)
> $\quad \mid \quad$ `simple` $=$ `false`
> **end**

   2. Greenwich mean sidereal time $[\mathrm{rad}]$.

> **if** `simple`
> $\quad \mid \quad \theta_{\mathrm{GMST}} = \mathtt{ut12gmst\_approx}(\mathrm{MJD}_{\mathrm{UT1}})$ (Algorithm **??**)
> **else**
> $\quad \mid \quad \theta_{\mathrm{GMST}} = \mathtt{ut12gmst}(\mathrm{MJD}_{\mathrm{UT1}})$ (Algorithm **??**)
> **end**

   3. Rotation matrix from ECI frame to ECEF frame.

$$\mathbf{R}_{\mathrm{eci}\to\mathrm{ecef}} = \mathbf{R}_3(\theta_{\mathrm{GMST}})$$

return • $\mathbf{R}_{\mathrm{eci}\to\mathrm{ecef}} \in \mathbb{R}^{3\times3}$ - rotation matrix from ECI frame to ECEF frame

---

Algorithm 102: `ecef2eci_matrix_approx`

Rotation matrix from ECEF frame to ECI frame (approximate).

Inputs:
   • $\mathrm{MJD}_{\mathrm{UT1}} \in \mathbb{R}$ - UT1 (Universal Time 1) $[\mathrm{MJD}]$
   • $\mathtt{simple} \in \mathbb{B}$ - *(OPTIONAL)* `true` if simpler calculation of GMST should be
                                 used, `false` otherwise (defaults to `false`)

Procedure:

1. Default `simple` to `false` if not input.

> **if** (`simple` not input)
> │　　`simple` = `false`
> **end**

2. Greenwich mean sidereal time [rad].

> **if** `simple`
> │　　$\theta_{\text{GMST}} = \texttt{ut12gmst\_approx}(\text{MJD}_{\text{UT1}})$　　　(Algorithm **??**)
> **else**
> │　　$\theta_{\text{GMST}} = \texttt{ut12gmst}(\text{MJD}_{\text{UT1}})$　　　(Algorithm **??**)
> **end**

3. Rotation matrix from ECEF frame to ECI frame.

$$\mathbf{R}_{\text{ecef}\to\text{eci}} = \mathbf{R}_3(-\theta_{\text{GMST}})$$

return • $\mathbf{R}_{\text{ecef}\to\text{eci}} \in \mathbb{R}^{3\times3}$　　- rotation matrix from ECEF frame to ECI frame

10.5　IAU2006/2000 Celestial-Terrestrial Transformations

10.5.1　Reference Frames

10.5.2　Earth Orientation Parameters

The following parameters are used to describe the orientation of the Earth:

- $x_p \in \mathbb{R}$ – polar coordinate of the CIP $['']$
- $y_p \in \mathbb{R}$ – polar coordinate of the CIP $['']$
- $dX \in \mathbb{R}$ – correction for X [mas]
- $dY \in \mathbb{R}$ – correction for Y [mas]
- LOD $\in \mathbb{R}$ – length of day [ms]

x_p and y_p are used to calculate the polar motion matrix (Section 10.5.4), while dX and dY are used in the calculation of the precession/nutation matrix (Section 10.5.6). LOD is used in determine the angular velocity of the Earth (Section 10.5.7). These parameters are reported daily by the IERS. We can obtain them from the very same file(s)/source(s) we got ΔUT1 from at the end of Section **??**:

- HTML version: https://datacenter.iers.org/data/html/finals2000A.data.html
- CSV version: https://datacenter.iers.org/data/csv/finals2000A.data.csv
- All versions[1]: https://www.iers.org/IERS/EN/DataProducts/EarthOrientationData/eop.html

　　　　The *Astrodynamics Toolbox* includes the function `eop_iau06` to obtain the Earth orientation parameters for a given time. This function is implemented as

---

[1]　To see all versions, navigate to the third link and click on the "`version metadata`" link next to "`finals.data (IAU2000)`" in the "`Standard EOP data files`" section of the "`Rapid data and predictions`" table. However, it is usually easiest to just use the `.csv` file provided by the second link.

```
[xp,yp,dX,dY,LOD] = eop_iau06(MJD_UT1,eop)
```

where `eop` is an $N \times 6$ array of Earth orientation parameters corresponding to the current time (which is input as `MJD_UT1`, the modified Julian date of UT1 (MJD_{UT1})). Each row of `eop` stores the Earth orientation parameters for a specific day. The columns of `eop` are organized as follows:

1. MJD – modified Julian date [MJD]
2. x_p – polar coordinate of the CIP ["]
3. y_p – polar coordinate of the CIP ["]
4. dX – correction term for X [mas]
5. dY – correction term for Y [mas]
6. LOD – length of day [ms]

To import `eop` to a MATLAB workspace, use

```
eop = load_eop
```

or

```
eop = load_eop(MJD_UTC0,duration)
```

For the second syntax, `MJD_UTC0` is the modified Julian date of UTC at the start of the simulation, and `duration` is the duration of the simulation. This second syntax only returns the portion of the full data set that is required for a simulation, which results in speed advantages due to passing less data to functions.

Note that the data file `eop.mat` is stored in the `toolbox/data/datafiles` folder, so this folder must be added to the current MATLAB path. To update `eop.mat`, run the `toolbox/data/update/update_eop.m` script.

10.5.3   Earth Rotation Angle and GMST

The **Earth rotation angle** (θ_{ERA}) is the angle between the Convention International Origin (CIO) and the Terrestrial Intermediate Origin (TIO). It is defined as [68, p. 52][103, pp. 212-213]

$$\theta_{\text{ERA}} = 2\pi \left[0.7790572732640 + 1.00273781191135448(\text{JD}_{\text{UT1}} - 2451545) \right] \tag{10.10}$$

Algorithm **??** implements Eq. (**??**) as a function of MJD_{UT1} and also wraps the ERA to the interval $[0, 2\pi)$.

Algorithm 103: `era_iau06`
Earth rotation angle (ERA) (IAU2006/2000, CIO based).

Inputs:
- $\text{MJD}_{\text{UT1}} \in \mathbb{R}$ - modified Julian date of UT1 (Universal Time 1) [MJD]

Procedure:
1. Julian date of UT1 [JD] (Algorithm 68).

$$\text{JD}_{\text{UT1}} = \texttt{mjd2jd}(\text{MJD}_{\text{UT1}})$$

2. ERA in radians.

$$\theta_{\text{ERA}} = 2\pi \left[0.7790572732640 + 1.00273781191135448(\text{JD}_{\text{UT1}} - 2451545) \right]$$

3. Wrap ERA to the interval $[0, 2\pi)$.

$$\theta_{\text{ERA}} = \text{mod}\left(\theta_{\text{ERA}}, 2\pi \right)$$

return • $\theta_{\text{ERA}} \in \mathbb{R}$ - Earth rotation angle (ERA) [rad]

The GMST, which we previously defined in Section **??**, can also be defined (in arcseconds) using ERA and the Julian centuries since 2000 of TT [68, p. 61][103, p. 216].

$$
\begin{aligned}
\theta''_{\mathrm{GMST}} = {} & \theta''_{\mathrm{ERA}} + 0.014506 + 4612.156534 T_{\mathrm{TT}} + 1.3915817 T_{\mathrm{TT}}^2 - 0.00000044 T_{\mathrm{TT}}^3 - 0.000029956 T_{\mathrm{TT}}^4 \\
& - 0.0000000368 T_{\mathrm{TT}}^5
\end{aligned}
$$

(10.11)

We do not need θ_{GMST} for the CIO based approach for the IAU2006/2000 implementation. Therefore, we want θ_{GMST} as a function of UT1 in this case as well (to match the operation of Algorithms **??** and **??**). To do this we introduce Algorithm 104.

Algorithm 104: `gmst_iau06`
Greenwich mean sidereal time (GMST) (IAU2006/2000, CIO based).

Inputs:
- $\mathrm{MJD}_{\mathrm{UT1}} \in \mathbb{R}$ - modified Julian date of UT1 (Universal Time 1) $[\mathrm{MJD}]$

Procedure:
1. Difference between UT1 and UTC $[\mathrm{s}]$ (see end of Section **??**).

    $\Delta\mathrm{UT1} = \texttt{get\_DUT1}(\mathrm{MJD}_{\mathrm{UT1}})$

2. Difference between TAI and UTC $[\mathrm{s}]$ (see end of Section **??**).

    $\Delta\mathrm{AT} = \texttt{get\_DAT}(\mathrm{MJD}_{\mathrm{UT1}})$

3. Modified Julian date of UTC (Algorithm 77).

    $\mathrm{MJD}_{\mathrm{UTC}} = \texttt{ut12utc}(\mathrm{MJD}_{\mathrm{UT1}}, \Delta\mathrm{UT1})$

4. Modified Julian date of TAI (Algorithm 78).

    $\mathrm{MJD}_{\mathrm{TAI}} = \texttt{utc2tai}(\mathrm{MJD}_{\mathrm{UTC}}, \Delta\mathrm{AT})$

5. Modified Julian date of TT (Algorithm 80).

    $\mathrm{MJD}_{\mathrm{TT}} = \texttt{tai2tt}(\mathrm{MJD}_{\mathrm{TAI}})$

6. Julian centuries since J2000.0 of TT $[\mathrm{c}]$ (Algorithm 70).

    $T_{\mathrm{TT}} = \texttt{mjd2t}(\mathrm{MJD}_{\mathrm{TT}})$

7. Earth rotation angle $[\mathrm{rad}]$ (Algorithm 103).

    $\theta_{\mathrm{ERA}} = \texttt{era\_iau06}(\mathrm{MJD}_{\mathrm{UT1}})$

8. Convert ERA to arcseconds (Algorithm 92).

    $\theta''_{\mathrm{ERA}} = \texttt{rad2arcsec}(\theta_{\mathrm{ERA}})$

9. GMST in arcseconds.

    $\begin{aligned}\theta''_{\mathrm{GMST}} = {} & \theta''_{\mathrm{ERA}} + 0.014506 + 4612.156534 T_{\mathrm{TT}} + 1.3915817 T_{\mathrm{TT}}^2 \\ & - 0.00000044 T_{\mathrm{TT}}^3 - 0.000029956 T_{\mathrm{TT}}^4 - 0.0000000368 T_{\mathrm{TT}}^5\end{aligned}$

10. Convert GMST to radians (Algorithm 93).

    $\theta_{\mathrm{GMST}} = \texttt{arcsec2rad}(\theta''_{\mathrm{GMST}})$

11. Wrap GMST to the interval $[0, 2\pi)$.

    $\theta_{\mathrm{GMST}} = \mathrm{mod}\,(\theta_{\mathrm{GMST}}, 2\pi)$

return • $\theta_{\mathrm{GMST}} \in \mathbb{R}$ - Greenwich mean sidereal time (GMST) $[\mathrm{rad}]$

10.5.4   Polar-Motion Matrix

The polar-motion matrix, $[\mathbf{W}]$, represents the transformation from the International Terrestrial Reference Frame (ITRF) to the Terrestrial Intermediate Reference System (TIRS). It is defined as

$$[\mathbf{W}] = \mathbf{R}_3(-s')\mathbf{R}_2(x_p)\mathbf{R}_1(y_p) \tag{10.12}$$

where s' (in arcseconds) is (currently[2]) defined as

$$s' = -(0.000047'')T_{\text{TT}} \tag{10.13}$$

and where x_p and y_p are the "polar coordinates" of the Celestial Intermediate Pole (CIP) given in arcseconds by the IERS Earth orientation data (see Section 10.5.2) [68, p. 48][103, p. 212].

Algorithm 105: `W_iau06`
Polar-motion matrix (IAU2006/2000, CIO based).

Inputs:
- $T_{\text{TT}} \in \mathbb{R}$ - Julian centuries since 2000 of TT (Terrestrial Time) $[c]$
- $x_p \in \mathbb{R}$ - polar coordinate of the CIP $['']$
- $y_p \in \mathbb{R}$ - polar coordinate of the CIP $['']$

Procedure:
1. s' $['']$

$$s' = -0.000047 T_{\text{TT}}$$

2. Convert s', x_p, and y_p to radians (Algorithm 93).

$$s' = \texttt{arcsec2rad}(s')$$
$$x_p = \texttt{arcsec2rad}(x_p)$$
$$y_p = \texttt{arcsec2rad}(y_p)$$

3. Sidereal-rotation matrix (Algorithms 1, 2, and 3).

$$[\mathbf{R}] = \texttt{rot3}(-s')\texttt{rot2}(x_p)\texttt{rot1}(y_p)$$

return • $[\mathbf{W}] \in \mathbb{R}^{3\times3}$ - polar-motion matrix (ITRF to TIRS)

10.5.5   Sidereal-Rotation Matrix

The sidereal-rotation matrix, $[\mathbf{R}]$, represents the transformation from the Terrestrial Intermediate Reference System (TIRS) to the Celestial Intermediate Reference System (CIRS). It is defined as [68, p. 48][103, pp. 212-213, 220]

$$[\mathbf{R}] = \mathbf{R}_3(-\theta_{\text{ERA}}) \tag{10.14}$$

---

[2]   The main component of s' gives

$$s' = -(-0.0015)\left(\frac{a_c^2}{1.2} + a_a^2\right)T_{\text{TT}}$$

Currently, the average values for a_c and a_a are $a_c = 0.26''$ and $a_a = 0.12''$ which provide the expression in Eq. (10.13).

Note that we can calculate θ_{ERA} from the Julian date of UT1 using Algorithm **??**. Thus, we define Algorithm 106 accordingly.

Algorithm 106: `R_iau06`

Sidereal-rotation matrix (IAU2006/2000, CIO based).

Inputs:

- $\text{MJD}_{\text{UT1}} \in \mathbb{R}$ - modified Julian date of UT1 (Universal Time 1) $[\text{MJD}]$

Procedure:

1. ERA $[\text{rad}]$ (Algorithm 103).

$$\theta_{\text{ERA}} = \texttt{era\_iau06}(\text{MJD}_{\text{UT1}})$$

2. Sidereal-rotation matrix (Algorithm 3).

$$[\mathbf{R}] = \texttt{rot3}(\theta_{\text{ERA}})$$

return • $[\mathbf{R}] \in \mathbb{R}^{3 \times 3}$ - sidereal-rotation matrix (TIRS to CIRS)

10.5.6 Precession-Nutation Matrix

The precession-nutation matrix, $[\mathbf{Q}]$, represents the transformation from the Celestial Intermediate Reference System (CIRS) to the Geocentric Celestial Reference Frame (GCRF). It is defined as [68, p. 49][103, pp. 213-215, 220]

$$[\mathbf{Q}] = \begin{bmatrix} 1 - aX^2 & -aXY & X \\ -aXY & 1 - aY^2 & Y \\ -X & -Y & 1 - a(X^2 + Y^2) \end{bmatrix} \mathbf{R}_3(s) \tag{10.15}$$

where

$$a \approx \frac{1}{2} + \frac{X^2 + Y^2}{8} \tag{10.16}$$

The calculations of X and Y are rather complex and depend on nested summations/calculations. The presentation is often confusing and is not conducive for easy implementation. The equations presented from here on out are compiled/adapted from [103, pp. 213–215] and [68, pp. 54–55, 59, 66–88]. Note that in some places, appropriate conversion factors have also been added so that all quantities are expressed in radians. The complete Algorithm for calculating $\mathbf{Q}$ is included as Algorithm 108 at the end of this section.

The fundamental arguments of nutation theory

Algorithm 107 below details the calculation of the fundamental arguments of nutation theory. Note that the portion of the equations given by [68, p. 67] that were originally in arcseconds are converted to degrees with the conversion factor $1/3600$.

Algorithm 107: `fund_arg_iau06`
Fundamental arguments for nutation theory (IAU2006/2000, CIO based).

Inputs:
- $T_{\mathrm{TT}} \in \mathbb{R}$ - Julian date since J2000.0 of TT (Terrestrial Time) [c]

Note:
- For the physical definitions/descriptions of F_1–F_6, see [68, p. 67].

Procedure:
1. Preallocate a vector $\mathbf{F} \in \mathbb{R}^{14}$ to store F_1 through F_{14}.
2. Delaunay variables [°].

$$F_1 = 134.96340251 + \frac{1}{3600}(1717915923.2178 T_{\mathrm{TT}} + 31.8792 T_{\mathrm{TT}}^2$$
$$+ \, 0.051635 T_{\mathrm{TT}}^3 - 0.00024470 T_{\mathrm{TT}}^4)$$

$$F_2 = 357.52910918 + \frac{1}{3600}(129596581.0481 T_{\mathrm{TT}} - 0.5532 T_{\mathrm{TT}}^2$$
$$- \, 0.000136 T_{\mathrm{TT}}^3 - 0.00001149 T_{\mathrm{TT}}^4)$$

$$F_3 = 93.27209062 + \frac{1}{3600}(1739527262.8478 T_{\mathrm{TT}} - 12.7512 T_{\mathrm{TT}}^2$$
$$+ \, 0.001037 T_{\mathrm{TT}}^3 + 0.00000417 T_{\mathrm{TT}}^4)$$

$$F_4 = 297.85019547 + \frac{1}{3600}(1602961601.2090 T_{\mathrm{TT}} - 6.3706 T_{\mathrm{TT}}^2$$
$$+ \, 0.006593 T_{\mathrm{TT}}^3 - 0.00003169 T_{\mathrm{TT}}^4)$$

$$F_5 = 125.04455501 + \frac{1}{3600}(-6962890.5431 T_{\mathrm{TT}} + 7.4722 T_{\mathrm{TT}}^2$$
$$+ \, 0.007702 T_{\mathrm{TT}}^3 - 0.00005939 T_{\mathrm{TT}}^4)$$

3. Convert the Delaunay elements to radians (Algorithm 88).

    for $i = 1$ **to** 5
        $F_i = \texttt{deg2rad}(F_i)$
    end

4. Mean heliocentric longitudes of the planets [rad].

$$F_6 = 4.402608842 + 2608.7903141574 T_{\mathrm{TT}}$$
$$F_7 = 3.176146697 + 1021.3285546211 T_{\mathrm{TT}}$$
$$F_8 = 1.753470314 + 628.3075849991 T_{\mathrm{TT}}$$
$$F_9 = 6.203480913 + 334.0612426700 T_{\mathrm{TT}}$$
$$F_{10} = 0.599546497 + 52.9690962641 T_{\mathrm{TT}}$$
$$F_{11} = 0.874016757 + 21.3299104960 T_{\mathrm{TT}}$$
$$F_{12} = 5.481293872 + 7.4781598567 T_{\mathrm{TT}}$$
$$F_{13} = 5.311886287 + 3.8133035638 T_{\mathrm{TT}}$$

5. General precession in longitude [rad].

$$F_{14} = 0.02438175 T_{\mathrm{TT}} + 0.00000538691 T_{\mathrm{TT}}^2$$

6. Wrap all parameters to the interval $[0, 2\pi)$.

> **for** $i = 1$ **to** 14
> $\quad\quad F_i = \mathrm{mod}\,(F_i, 2\pi)$
> **end**

return • $\mathbf{F} \in \mathbb{R}^{14}$ - fundamental arguments for nutation theory

Required data

To calculate X, Y, and s, we need to process some data. The following are links from which we can download the data:

- X data (`tab.2a.txt`): https://iers-conventions.obspm.fr/content/chapter5/additional_info/tab5.2a.txt
- Y data (`tab.2b.txt`): https://iers-conventions.obspm.fr/content/chapter5/additional_info/tab5.2b.txt
- s data (`tab.2d.txt`): https://iers-conventions.obspm.fr/content/chapter5/additional_info/tab5.2d.txt

The X data is in the following form [68, p. 55]:

i	$a_{x,i}$ [μas]	$b_{x,i}$ [μas]	$M_{i,1}$	$M_{i,2}$	$\cdots$	$M_{i,14}$
1	-6844318.44	1328.67	0	0	$\cdots$	0
2	-523908.04	-544.75	0	0	$\cdots$	0
3	-90552.22	111.23	0	0	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
1600	-0.10	-0.02	0	0	$\cdots$	0

The Y data is in the following form [68, p. 55]:

i	$a_{y,i}$ [μas]	$b_{y,i}$ [μas]	$N_{i,1}$	$N_{i,2}$	$\cdots$	$N_{i,14}$
1	1538.18	9205236.26	0	0	$\cdots$	0
2	-458.66	573033.42	0	0	$\cdots$	0
3	137.41	97846.69	0	0	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
1275	-0.02	0.11	0	0	$\cdots$	0

The s data is in the following form [68, p. 55]:

i	$a_{s,i}$ [µas]	$b_{s,i}$ [µas]	$O_{i,1}$	$O_{i,2}$	$\cdots$	$O_{i,14}$
1	-2640.73	0.39	0	0	$\cdots$	0
2	-63.53	0.02	0	0	$\cdots$	0
3	-11.75	-0.01	0	0	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
66	-0.26	-0.01	0	0	$\cdots$	0

During initial data processing, we can form the vectors $\mathbf{a}_x$, $\mathbf{a}_y$, $\mathbf{a}_s$, $\mathbf{b}_x$, $\mathbf{b}_y$, $\mathbf{b}_s$ and the matrices $\mathbf{M}$, $\mathbf{N}$, and $\mathbf{O}$ from the data in these tables[3]. It is easiest to store all these vectors/matrices in a structure, save that structure to a data file, and load the data file during the initialization of an orbit propagator. **It is also essential to convert the elements of $\mathbf{a}_x$, $\mathbf{a}_y$, $\mathbf{a}_s$, $\mathbf{b}_x$, $\mathbf{b}_y$, $\mathbf{b}_s$ to radians**. The data tables give theses quantities in units of arcseconds [68, p. 55], but all calculations in this section assume these quantities are in radians. It is easiest to handle this unit conversion during initial data processing as opposed to during each execution of `Q_iau06` (Algorithm 108).

This initial data processing is handled by the `toolbox/data/create/create_XYs_iau06.m` script, which saves a structure, `XYs_iau06`, to the data file `XYs_iau06.mat` located in the `toolbox/data/datafiles` folder. This structure stores the vectors $\mathbf{a}_x$, $\mathbf{a}_y$, $\mathbf{a}_s$, $\mathbf{b}_x$, $\mathbf{b}_y$, $\mathbf{b}_s$ and the matrices $\mathbf{M}$, $\mathbf{N}$, and $\mathbf{O}$. To import `XYs_iau06` to a MATLAB workspace, use

```
XYs_iau06 = load_XYs_iau06
```

Calculating X

The 0th term is given by

$$
\boxed{
\begin{aligned}
X_0 = \left(\frac{\pi}{648000}\right) & \left[-0.016617 + 2004.191898 T_{\text{TT}} - 0.4297829 T_{\text{TT}}^2 - 0.19861834 T_{\text{TT}}^3 \right. \\
& \left. + 0.000007578 T_{\text{TT}}^4 + 0.0000059285 T_{\text{TT}}^5\right]
\end{aligned}
}
\tag{10.17}
$$

To calculate terms 1–5, we need to calculate f_i at each iteration as

$$
\boxed{f_i = \sum_{j=1}^{14} N_{ij} F_j}
\tag{10.18}
$$

Then we can write

$$
\boxed{
\begin{aligned}
X_1 &= \sum_{i=1}^{1306} \left[a_{x,i}\sin\left(f_i\right) + b_{x,i}\cos\left(f_i\right)\right] \\
X_2 &= T_{\text{TT}} \sum_{i=1307}^{1559} \left[a_{x,i}\sin\left(f_i\right) + b_{x,i}\cos\left(f_i\right)\right] \\
X_3 &= T_{\text{TT}}^2 \sum_{i=1560}^{1595} \left[a_{x,i}\sin\left(f_i\right) + b_{x,i}\cos\left(f_i\right)\right] \\
X_4 &= T_{\text{TT}}^3 \sum_{i=1596}^{1599} \left[a_{x,i}\sin\left(f_i\right) + b_{x,i}\cos\left(f_i\right)\right] \\
X_5 &= T_{\text{TT}}^4 \left[a_{x,1600}\cos\left(f_{1600}\right) + b_{x,1600}\sin\left(f_{1600}\right)\right]
\end{aligned}
}
\tag{10.19}
$$

---

[3]   Note that all the M_{ij}, N_{ij}, and O_{ij} data in the excerpts above is 0, but this is generally *not* the case. It just happened that these excerpts didn't have any nonzero elements.

$$\boxed{X = X_0 + X_1 + X_2 + X_3 + X_4 + X_5 + dX} \tag{10.20}$$

Calculating Y

The 0th term is given by

$$\boxed{\begin{aligned} Y_0 = \left(\frac{\pi}{648000}\right) &\left[-0.006951 - 0.025896 T_{\text{TT}} - 22.4072747 T_{\text{TT}}^2 + 0.00190059 T_{\text{TT}}^3 \right. \\ &\left. + 0.001112526 T_{\text{TT}}^4 + 0.0000001358 T_{\text{TT}}^5\right] \end{aligned}} \tag{10.21}$$

To calculate terms 1–5, we need to calculate g_i at each iteration as

$$\boxed{g_i = \sum_{j=1}^{14} M_{ij} F_j} \tag{10.22}$$

Then we have

$$\boxed{\begin{aligned} Y_1 &= \sum_{i=1}^{962} \left[a_{y,i} \sin\left(g_i\right) + b_{y,i} \cos\left(g_i\right)\right] \\ Y_2 &= T_{\text{TT}} \sum_{i=963}^{1239} \left[a_{y,i} \sin\left(g_i\right) + b_{y,i} \cos\left(g_i\right)\right] \\ Y_3 &= T_{\text{TT}}^2 \sum_{i=1240}^{1269} \left[a_{y,i} \sin\left(g_i\right) + b_{y,i} \cos\left(g_i\right)\right] \\ Y_4 &= T_{\text{TT}}^3 \sum_{i=1270}^{1274} \left[a_{y,i} \sin\left(g_i\right) + b_{y,i} \cos\left(g_i\right)\right] \\ Y_5 &= T_{\text{TT}}^4 \left[a_{y,1275} \cos\left(f_{1275}\right) + b_{y,1275} \sin\left(f_{1275}\right)\right] \end{aligned}} \tag{10.23}$$

$$\boxed{Y = Y_0 + Y_1 + Y_2 + Y_3 + Y_4 + Y_5 + dY} \tag{10.24}$$

Calculating s

The 0th term is given by

$$\boxed{\begin{aligned} s_0 = \left(\frac{\pi}{648000}\right) &\left[0.000094 + 0.00380865 T_{\text{TT}} - 0.00012268 T_{\text{TT}}^2 - 0.07257411 T_{\text{TT}}^3 \right. \\ &\left. + 0.00002798 T_{\text{TT}}^4 + 0.00001562 T_{\text{TT}}^5\right] \end{aligned}} \tag{10.25}$$

To calculate terms 1–5, we need to calculate h_i at each iteration as

$$\boxed{h_i = \sum_{j=1}^{14} O_{ij} F_j} \tag{10.26}$$

Then we have

$$s_1 = \sum_{i=1}^{33} [a_{s,i} \sin(h_i) + b_{s,i} \cos(h_i)]$$

$$s_2 = T_{\text{TT}} \sum_{i=34}^{36} [a_{s,i} \sin(h_i) + b_{s,i} \cos(h_i)]$$

$$s_3 = T_{\text{TT}}^2 \sum_{i=37}^{61} [a_{s,i} \sin(h_i) + b_{s,i} \cos(h_i)] \qquad (10.27)$$

$$s_4 = T_{\text{TT}}^3 \sum_{i=62}^{65} [a_{s,i} \sin(h_i) + b_{s,i} \cos(h_i)]$$

$$s_5 = T_{\text{TT}}^4 [a_{s,66} \cos(f_{66}) + b_{s,66} \sin(f_{66})]$$

$$s = -\frac{XY}{2} + s_0 + s_1 + s_2 + s_3 + s_4 + s_5 \qquad (10.28)$$

Algorithm for computing Q

Algorithm 108: Q_iau06

Precession-nutation matrix (IAU2006/2000, CIO based).

Inputs:
- $T_{\text{TT}} \in \mathbb{R}$ - Julian date since J2000.0 of TT (Terrestrial Time) [c]
- $dX \in \mathbb{R}$ - correction term for X [mas]
- $dY \in \mathbb{R}$ - correction term for Y [mas]
- $\boldsymbol{a}_x, \boldsymbol{b}_x \in \mathbb{R}^{1600}$ - coefficients for X [rad]
- $\boldsymbol{a}_y, \boldsymbol{b}_y \in \mathbb{R}^{1275}$ - coefficients for Y [rad]
- $\boldsymbol{a}_s, \boldsymbol{b}_s \in \mathbb{R}^{66}$ - coefficients for s [rad]
- $\mathbf{N} \in \mathbb{R}^{1600 \times 14}$ - fundamental argument coefficients for X
- $\mathbf{M} \in \mathbb{R}^{1275 \times 14}$ - fundamental argument coefficients for Y
- $\mathbf{O} \in \mathbb{R}^{66 \times 14}$ - fundamental argument coefficients for s

Procedure:
1. Obtain the fundamental arguments for nutation theory (Algorithm 107) [rad].

   $$\mathbf{F} = \texttt{fund\_arg\_iau06}(T_{\text{TT}})$$

2. Convert dX and dY to arcseconds.

   $$dX = 0.001 dX$$
   $$dY = 0.001 dY$$

3. Convert dX and dY to radians (Algorithm 93).

   $$dX = \texttt{arcsec2rad}(dX)$$
   $$dY = \texttt{arcsec2rad}(dY)$$

4. Calculate X [rad]. Note that the summations should be evaluated using for loops, and f_i should be recalculated once every iteration within a for loop as $f_i =$

$\sum_{j=1}^{14} N_{ij} F_j.$

$$X_0 = \left(\frac{\pi}{648000}\right) \left[-0.016617 + 2004.191898 T_{\mathrm{TT}} - 0.4297829 T_{\mathrm{TT}}^2\right.$$
$$\left. - 0.19861834 T_{\mathrm{TT}}^3 + 0.000007578 T_{\mathrm{TT}}^4 + 0.0000059285 T_{\mathrm{TT}}^5\right]$$

$$X_1 = \sum_{i=1}^{1306} \left[a_{x,i} \sin\left(f_i\right) + b_{x,i} \cos\left(f_i\right)\right]$$

$$X_2 = T_{\mathrm{TT}} \sum_{i=1307}^{1559} \left[a_{x,i} \sin\left(f_i\right) + b_{x,i} \cos\left(f_i\right)\right]$$

$$X_3 = T_{\mathrm{TT}}^2 \sum_{i=1560}^{1595} \left[a_{x,i} \sin\left(f_i\right) + b_{x,i} \cos\left(f_i\right)\right]$$

$$X_4 = T_{\mathrm{TT}}^3 \sum_{i=1595}^{1599} \left[a_{x,i} \sin\left(f_i\right) + b_{x,i} \cos\left(f_i\right)\right]$$

$$X_5 = T_{\mathrm{TT}}^4 \left[a_{x,1600} \cos\left(f_{1600}\right) + b_{x,1600} \sin\left(f_{1600}\right)\right]$$

$$X = X_0 + X_1 + X_2 + X_3 + X_4 + X_5 + dX$$

5. Calculate Y [rad]. Note that the summations should be evaluated using for loops, and g_i should be recalculated once every iteration within a for loop as $g_i = \sum_{j=1}^{14} M_{ij} F_j$.

$$Y_0 = \left(\frac{\pi}{648000}\right) \left[-0.006951 - 0.025896 T_{\mathrm{TT}} - 22.4072747 T_{\mathrm{TT}}^2\right.$$
$$\left. + 0.00190059 T_{\mathrm{TT}}^3 + 0.001112526 T_{\mathrm{TT}}^4 + 0.0000001358 T_{\mathrm{TT}}^5\right]$$

$$Y_1 = \sum_{i=1}^{962} \left[a_{y,i} \sin\left(g_i\right) + b_{y,i} \cos\left(g_i\right)\right]$$

$$Y_2 = T_{\mathrm{TT}} \sum_{i=963}^{1239} \left[a_{y,i} \sin\left(g_i\right) + b_{y,i} \cos\left(g_i\right)\right]$$

$$Y_3 = T_{\mathrm{TT}}^2 \sum_{i=1240}^{1269} \left[a_{y,i} \sin\left(g_i\right) + b_{y,i} \cos\left(g_i\right)\right]$$

$$Y_4 = T_{\mathrm{TT}}^3 \sum_{i=1270}^{1274} \left[a_{y,i} \sin\left(g_i\right) + b_{y,i} \cos\left(g_i\right)\right]$$

$$Y_5 = T_{\mathrm{TT}}^4 \left[a_{y,1275} \cos\left(f_{1275}\right) + b_{y,1275} \sin\left(f_{1275}\right)\right]$$

$$Y = Y_0 + Y_1 + Y_2 + Y_3 + Y_4 + Y_5 + dY$$

6. Calculate s [rad]. Note that the summations should be evaluated using for loops, and h_i should be recalculated once every iteration within a for loop as $h_i =$

$$\sum_{j=1}^{14} O_{ij} F_j.$$

$$s_0 = \left(\frac{\pi}{648000}\right) \big[0.000094 + 0.00380865 T_{\mathrm{TT}} - 0.00012268 T_{\mathrm{TT}}^2$$
$$- 0.07257411 T_{\mathrm{TT}}^3 + 0.00002798 T_{\mathrm{TT}}^4 + 0.00001562 T_{\mathrm{TT}}^5\big]$$

$$s_1 = \sum_{i=1}^{33} \left[a_{s,i} \sin\left(h_i\right) + b_{s,i} \cos\left(h_i\right)\right]$$

$$s_2 = T_{\mathrm{TT}} \sum_{i=34}^{36} \left[a_{s,i} \sin\left(h_i\right) + b_{s,i} \cos\left(h_i\right)\right]$$

$$s_3 = T_{\mathrm{TT}}^2 \sum_{i=37}^{61} \left[a_{s,i} \sin\left(h_i\right) + b_{s,i} \cos\left(h_i\right)\right]$$

$$s_4 = T_{\mathrm{TT}}^3 \sum_{i=62}^{65} \left[a_{s,i} \sin\left(h_i\right) + b_{s,i} \cos\left(h_i\right)\right]$$

$$s_5 = T_{\mathrm{TT}}^4 \left[a_{s,66} \cos\left(f_{66}\right) + b_{s,66} \sin\left(f_{66}\right)\right]$$

$$s = -\frac{XY}{2} + s_0 + s_1 + s_2 + s_3 + s_4 + s_5$$

7. a [rad]

$$a \approx \frac{1}{2} + \frac{X^2 + Y^2}{8}$$

8. Precession-nutation matrix (Algorithm 3).

$$[\mathbf{Q}] = \begin{bmatrix} 1 - aX^2 & -aXY & X \\ -aXY & 1 - aY^2 & Y \\ -X & -Y & 1 - a(X^2 + Y^2) \end{bmatrix} \texttt{rot3}(s)$$

return • $[\mathbf{Q}] \in \mathbb{R}^{3\times3}$ - precession-nutation matrix (CIRS to GCRF)

10.5.7 Angular Velocity

The rotation rate of the Earth (in $\mathrm{rad/s}$) is given by

$$\omega_\oplus = (7.292115146706979 \times 10^{-5}) \left(1 - \frac{\mathrm{LOD}^s}{86400}\right) \tag{10.29}$$

where the **length of day (LOD)** is the instantaneous rate of change of UT1 with respect to a uniform time scale (i.e. UTC or TAI). Although not explicitly stated, the equations in [103, pp. 220, 222] imply that the angular velocity of the Earth is resolved in the TIRS (Terrestrial Intermediate Reference System) frame. Thus,

$$[\boldsymbol{\omega}_\oplus]_{\mathrm{tirs}} = \begin{bmatrix} 0 \\ 0 \\ \omega_\oplus \end{bmatrix} \tag{10.30}$$

We generally want the angular velocity of the Earth resolved in a celestial reference frame, which in the case of the IAU2006/2000 conventions is the GCRF (Geocentric Celestial Reference Frame). We know that the sidereal-rotation

matrix, $[\mathbf{R}]$, maps vectors from the TIRS frame to the CIRS frame, and that the precession-nutation matrix, $[\mathbf{Q}]$, maps vectors from the CIRS frame to the GCRF. Therefore,

$$\boxed{[\boldsymbol{\omega}_\oplus]_{\text{gcrf}} = [\mathbf{Q}][\mathbf{R}][\boldsymbol{\omega}]_{\text{tirs}}} \tag{10.31}$$

Algorithm 109: `w_earth_iau06`

Angular velocity of the Earth resolved in the ITRF frame (IAU2006/2000, CIO based).

Inputs:
- $\text{LOD} \in \mathbb{R}$ - length of day [ms]
- $[\mathbf{Q}] \in \mathbb{R}^{3\times3}$ - precession-nutation matrix
- $[\mathbf{R}] \in \mathbb{R}^{3\times3}$ - sidereal-rotation matrix

Procedure:
1. Convert length of day to seconds.

$$\text{LOD} = (0.001)\text{LOD}$$

2. Rotation rate of the Earth [rad/s].

$$\omega_\oplus = (7.292115146706979 \times 10^{-5})\left(1 - \frac{\text{LOD}}{86400}\right)$$

3. Angular velocity of the Earth resolved in the TIRS frame [rad/s].

$$[\boldsymbol{\omega}_\oplus]_{\text{tirs}} = \begin{bmatrix} 0 \\ 0 \\ \omega_\oplus \end{bmatrix}$$

4. Angular velocity of the Earth resolved in the GCRF [rad/s].

$$[\boldsymbol{\omega}_\oplus]_{\text{gcrf}} = [\mathbf{Q}][\mathbf{R}][\boldsymbol{\omega}]_{\text{tirs}}$$

return • $[\boldsymbol{\omega}_\oplus]_{\text{gcrf}} \in \mathbb{R}^3$ - Earth angular velocity resolved in GCRF [rad/s]

10.5.8 Overall IAU2006/2000 Calculation Procedure

Algorithm 110: `iau06`

Rotation matrices from the IAU2006/2000 CIO based theory.

Inputs:

- $\mathrm{MJD}_{\mathrm{UT1}} \in \mathbb{R}$ - modified Julian date of UT1 (Universal Time 1) [MJD]
- $\mathrm{MJD}_{\mathrm{TT}} \in \mathbb{R}$ - modified Julian date of TT (Terrestrial Time) [MJD]
- $x_p \in \mathbb{R}$ - polar coordinate of the CIP $['']$
- $y_p \in \mathbb{R}$ - polar coordinate of the CIP $['']$
- $dX \in \mathbb{R}$ - correction term for X [mas]
- $dY \in \mathbb{R}$ - correction term for Y [mas]
- $\mathrm{LOD} \in \mathbb{R}$ - length of day [ms]
- $\boldsymbol{a}_x, \boldsymbol{b}_x \in \mathbb{R}^{1600}$ - coefficients for X [rad]
- $\boldsymbol{a}_y, \boldsymbol{b}_y \in \mathbb{R}^{1275}$ - coefficients for Y [rad]
- $\boldsymbol{a}_s, \boldsymbol{b}_s \in \mathbb{R}^{66}$ - coefficients for s [rad]
- $\mathbf{N} \in \mathbb{R}^{1600 \times 14}$ - fundamental argument coefficients for X
- $\mathbf{M} \in \mathbb{R}^{1275 \times 14}$ - fundamental argument coefficients for Y
- $\mathbf{O} \in \mathbb{R}^{66 \times 14}$ - fundamental argument coefficients for s

Procedure:

1. Julian centuries since J2000.0 of TT [] (Algorithm 70).

$$T_{\mathrm{TT}} = \mathtt{mjd2t}(\mathrm{MJD}_{\mathrm{TT}})$$

2. Precession-nutation matrix (Algorithm 108).

$$[\mathbf{Q}] = \mathtt{Q\_iau06}(T_{\mathrm{TT}}, dX, dY, \mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_s, \mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_s, \mathbf{M}, \mathbf{N}, \mathbf{O})$$

3. Sidereal-rotation matrix (Algorithm 106).

$$[\mathbf{R}] = \mathtt{R\_iau06}(\mathrm{MJD}_{\mathrm{UT1}})$$

4. Polar-motion matrix (Algorithm 105).

$$[\mathbf{W}] = \mathtt{W\_iau06}(T_{\mathrm{TT}}, x_p, y_p)$$

5. Rotation matrix from ITRF to GCRF.

$$\mathbf{R}_{\mathrm{itrf} \rightarrow \mathrm{gcrf}} = [\mathbf{Q}][\mathbf{R}][\mathbf{W}]$$

6. Rotation matrix from GCRF to ITRF.

$$\mathbf{R}_{\mathrm{gcrf} \rightarrow \mathrm{itrf}} = \mathbf{R}_{\mathrm{itrf} \rightarrow \mathrm{gcrf}}^{T}$$

7. Angular velocity of the Earth resolved in the GCRF [rad/s] (Algorithm 109).

$$[\boldsymbol{\omega}]_{\mathrm{gcrf}} = \mathtt{w\_earth\_iau06}(\mathrm{LOD}, [\mathbf{Q}], [\mathbf{R}])$$

Outputs:
- $\mathbf{R}_{\mathrm{itrf} \rightarrow \mathrm{gcrf}} \in \mathbb{R}^{3 \times 3}$ - passive rotation matrix from International Terrestrial
Reference Frame (ITRF) to Geocentric Celestial
Reference Frame (GCRF)
- $\mathbf{R}_{\mathrm{gcrf} \rightarrow \mathrm{itrf}} \in \mathbb{R}^{3 \times 3}$ - passive rotation matrix from Geocentric Celestial
Reference Frame (GCRF) to International Terrestrial
Reference Frame (ITRF)
- $[\boldsymbol{\omega}_{\oplus}]_{\mathrm{gcrf}} \in \mathbb{R}^3$ - Earth angular velocity resolved in GCRF [rad]
- $[\mathbf{Q}] \in \mathbb{R}^{3 \times 3}$ - precession-nutation matrix (CIRS to GCRF)
- $[\mathbf{R}] \in \mathbb{R}^{3 \times 3}$ - sidereal-rotation matrix (TIRS to CIRS)
- $[\mathbf{W}] \in \mathbb{R}^{3 \times 3}$ - polar-motion matrix (ITRF to TIRS)

10.6   Reference Ellipsoids

The **polar radius**, R_{pole}, can be calculated as

$$\boxed{R_{\text{pole}} = R_{\oplus}(1 - f)} \tag{10.32}$$

where $R_{\oplus}$ is the mean equatorial radius, and where the **flattening**, f, is defined as [60, p. 187]

$$\boxed{f = \frac{R_{\oplus} - R_{\text{pole}}}{R_{\oplus}}} \tag{10.33}$$

The **eccentricity** of the Earth, $e_{\oplus}$, is defined as [**eccentricity_mathematics_wikipedia**]

$$\boxed{e_{\oplus} = \sqrt{1 - \left(\frac{R_{\text{pole}}}{R_{\oplus}}\right)^2}} \tag{10.34}$$

Substituting Eq. (10.32) into Eq. (10.34) [60, p. 187],

$$e_{\oplus} = \sqrt{1 - (1 - f)^2} \tag{10.35}$$

This can be simplified to [**flattening_wikipedia**]

$$\boxed{e_{\oplus} = \sqrt{f(2 - f)}} \tag{10.36}$$

We can also obtain a useful relationship between $e_{\oplus}$ and f by squaring both sides of Eq. (10.35) and rearranging slightly and solving for $(1 - f)^2$:

$$\boxed{(1 - f)^2 = 1 - e_{\oplus}^2} \tag{10.37}$$

From this it follows that

$$1 - f = \sqrt{1 - e_{\oplus}^2} \tag{10.38}$$

Substituting Eq. (10.38), we can also write the polar radius as [**damico_279a_lecture_7**]

$$\boxed{R_{\text{pole}} = R_{\oplus}\sqrt{1 - e_{\oplus}^2}} \tag{10.39}$$

Finally, we can also solve Eq. (10.37) for f.

$$\boxed{f = 1 - \sqrt{1 - e_{\oplus}^2}} \tag{10.40}$$

10.7   Planetographic (Geographic) Coordinates

10.7.1   Planetodetic (Geodetic) Coordinates

The **geodetic longitude**, λ, satisfies the relation [60, p. 188]

$$\tan \lambda = \frac{r_Y}{r_X}$$

We define λ to be contained in the interval $[-180°, 180°]$. Since we must find ϕ_{gc} in the correct quadrant, we need to use the four-quadrant inverse tangent when solving the equation above for λ.

$$\boxed{\lambda = \arctan2\left(r_Y, r_X\right)} \tag{10.41}$$

Since arctan2 already returns angles in the interval $[-180°, 180°]$, we do not have to "wrap" the result of Eq. (10.41) to another interval.

To obtain the geodetic coordinates corresponding to an ECEF position, we can use Algorithm 111 below, adapted from [**damico_279a_lecture_7**]. The last step of this algorithm is taken from [103, p. 172], which suggests using

$$h = \frac{r_Z}{\sin \phi} - S_\oplus$$

when within $1°$ of the poles, where $S_\oplus$ is defined as [103, p. 138]

$$S_\oplus = \frac{R_\oplus (1 - e_\oplus^2)}{\sqrt{1 - e_\oplus^2 \sin^2 \phi}}$$

However, we also know that [**damico_279a_lecture_7**][60, p. 188][103, p. 172]

$$\boxed{N_\phi = \frac{R_\oplus}{\sqrt{1 - e_\oplus^2 \sin^2 \phi}}} \tag{10.42}$$

Thus, we have $S_\oplus = N_\phi (1 - e_\oplus^2)$ from which it follows that

$$h = \frac{r_Z}{\sin \phi} - N_\phi (1 - e_\oplus^2)$$

Algorithm 111: `ecef2geod`
Geodetic coordinates from the ECEF position vector.

Inputs:
- $[\mathbf{r}]_{\text{ecef}} \in \mathbb{R}^3$ - satellite position resolved in ECEF frame $[\text{m}]$

Procedure:
1. Define $R_\oplus$ $[\text{m}]$ and $e_\oplus$ using the values from Appendix **??**.
2. Extract r_X, r_Y, and r_Z from $[\mathbf{r}]_{\text{ecef}} = (r_X, r_Y, r_Z)^T$.
3. Satellite position magnitude $[\text{m}]$.

$$r = \|[\mathbf{r}]_{\text{ecef}}\|$$

4. Scalar projection of the satellite ECEF position vector onto the XY plane of the ECEF frame $[\text{m}]$.

$$r_{XY} = \sqrt{r_X^2 + r_Y^2}$$

5. Geodetic longitude $[°]$.

$$\lambda = \arctan2 (r_Y, r_X)$$

6. Initial guess for geodetic latitude $[°]$.

$$\phi_{\text{old}} = \arcsin \left(\frac{r_Z}{r} \right)$$

7. Set tolerance (TOL) and maximum number of iterations ($k_{\max}$).

$$\text{TOL} = 10^{-12}$$
$$k_{\max} = 200$$

8. Initialize the error so the loop will be entered.

$$\varepsilon = (2)(\text{TOL})$$

9. Fixed-point iteration to solve for geodetic latitude.

$k = 1$

while $\varepsilon > \text{TOL}$ **and** $i < k_{\max}$

    (a) Radius of curvature in the meridian $[\text{m}]$.

$$N_\phi = \frac{R_\oplus}{\sqrt{1 - e_\oplus^2 \sin^2 \phi_{\text{old}}}}$$

    (b) Update estimate of geodetic latitude $[°]$.

$$\phi_{\text{new}} = \text{arctan2} \left[r_Z + N_\phi e_\oplus^2 \sin\left(\phi_{\text{old}}\right), r_{XY} \right]$$

    (c) Store the current estimate for the next iteration.

$$\phi_{\text{old}} = \phi_{\text{new}}$$

    (d) Increment loop index.

$$k = k + 1$$

end

10. Geodetic latitude $[°]$.

$$\phi = \phi_{\text{new}}$$

11. Geodetic altitude $[\text{m}]$.

if $(89° \leq |\phi| \leq 90°)$

$$h = \frac{r_Z}{\sin\left(\phi\right)} - N_\phi(1 - e_\oplus^2)$$

else

$$h = \frac{r_{XY}}{\cos\left(\phi\right)} - N_\phi$$

end

return
- $\phi \in \mathbb{R}$ - geodetic latitude $[°]$
- $\lambda \in \mathbb{R}$ - geodetic longitude $[°]$
- $h \in \mathbb{R}$ - geodetic altitude $[\text{m}]$

To obtain the ECEF position corresponding to a set of geodetic coordinates, [60, p. 188] gives

$$[\mathbf{r}]_{\text{ecef}} = \begin{bmatrix} (N_\phi + h) \cos\phi \cos\lambda \\ (N_\phi + h) \cos\phi \sin\lambda \\ \left[(1 - f)^2 N_\phi + h\right] \sin\phi \end{bmatrix}$$

From Eq. (**??**) (Section **??**), we have

$$(1 - f)^2 = 1 - e_\oplus^2$$

Thus, we can write $[\mathbf{r}]_{\text{ecef}}$ as

$$[\mathbf{r}]_{\text{ecef}} = \begin{bmatrix} (N_\phi + h) \cos \phi \cos \lambda \\ (N_\phi + h) \cos \phi \sin \lambda \\ \left[\left(1 - e_\oplus^2 \right) N_\phi + h \right] \sin \phi \end{bmatrix} \tag{10.43}$$

Algorithm 112: `geod2ecef`
ECEF position vector from the geodetic coordinates.

Inputs:
- $\phi \in \mathbb{R}$ - geodetic latitude $[°]$
- $\lambda \in \mathbb{R}$ - geodetic longitude $[°]$
- $h \in \mathbb{R}$ - geodetic altitude $[\text{m}]$

Procedure:
1. Define $R_\oplus$ $[\text{m}]$ and $e_\oplus$ using the values from Appendix **??**.
2. Radius of curvature in the meridian $[\text{m}]$.

$$N_\phi = \frac{R_\oplus}{\sqrt{1 - e_\oplus^2 \sin^2 \phi}}$$

3. ECEF position vector $[\text{m}]$.

$$[\mathbf{r}]_{\text{ecef}} = \begin{bmatrix} (N_\phi + h) \cos \phi \cos \lambda \\ (N_\phi + h) \cos \phi \sin \lambda \\ \left[\left(1 - e_\oplus^2 \right) N_\phi + h \right] \sin \phi \end{bmatrix}$$

Outputs:
- $[\mathbf{r}]_{\text{ecef}} \in \mathbb{R}^3$ - satellite position resolved in ECEF frame $[\text{m}]$

Approximation of planetodetic altitude

A good approximation for the geodetic altitude, h, is given by

$$h = r - r_s \tag{10.44}$$

where r is the magnitude of the satellite position vector and r_s is the radius of the Earth at the subsatellite position. r_s can be approximated as

$$r_s \approx \frac{R_\oplus(1 - f)}{\sqrt{1 - (2f - f^2) \cos^2 \delta}} = \frac{R_\oplus(1 - f)}{\sqrt{1 - f(2 - f) \cos^2 \delta}} \tag{10.45}$$

where $R_\oplus$ is the mean equatorial radius of the Earth, f is the Earth's flattening coefficient, and δ is the declination of the satellite. It is assumed that δ equals the geocentric latitude of the subsatellite point, ϕ_{gc} [**gtds_1989**].

$$\delta \approx \phi_{\text{gc}} \tag{10.46}$$

Substituting Eq. (10.46) into Eq. (10.45), then substituting that result into Eq. (10.44), we get

$$h \approx r - \frac{R_\oplus(1 - f)}{\sqrt{1 - f(2 - f) \cos^2 \phi_{\text{gc}}}} \tag{10.47}$$

Squaring both sides of Eq. (**??**),

$$e_\oplus^2 = f(2 - f)$$

Substituting this into Eq. (10.47),

$$h \approx r - \frac{R_\oplus(1 - f)}{\sqrt{1 - e_\oplus^2 \cos^2 \phi_{\text{gc}}}}$$

From Eq. (**??**), we also know that the polar radius is defined as

$$R_{\text{pole}} = R_\oplus(1 - f)$$

Substituting this into our expression for h,

$$h \approx r - \frac{R_{\text{pole}}}{\sqrt{1 - e_\oplus^2 \cos^2 \phi_{\text{gc}}}}$$

We also know from Eq. (**??**) that

$$\phi_{\text{gc}} = \arcsin\left(\frac{r_z}{r}\right)$$

where $[\mathbf{r}]_{\text{ecef}} = (r_X, r_Y, r_Z)^T$ is the position resolved in the ECEF frame and where $r = \|\mathbf{r}\| = \sqrt{r_X^2 + r_Y^2 + r_Z^2}$. From the equation for ϕ_{gc}, we know that

$$\sin \phi_{gc} = \frac{r_z}{r}$$

Let's examine the geometry of the corresponding triangle, shown in Fig. 10.1. We can see that
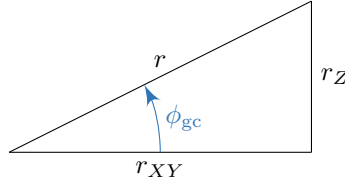
Figure 10.1: Geometry of the geocentric latitude.

$$\cos \phi_{\text{gc}} = \frac{r_{XY}}{r}$$

where

$$\boxed{r_{XY} = \sqrt{r_X^2 + r_Y^2}} \qquad (10.48)$$

is the projection[4] of $[\mathbf{r}]_{\text{ecef}}$ onto the XY-plane. It follows that

$$\boxed{h \approx r - \frac{R_{\text{pole}}}{\sqrt{1 - e_\oplus^2 \left(\frac{r_{XY}}{r}\right)^2}}} \qquad (10.49)$$

---

[4]   We can also find this equation from the Pythagorean theorem:

$$r_{XY}^2 + r_Z^2 = r^2 \quad \rightarrow \quad r_{XY}^2 = \left(\sqrt{r_X^2 + r_Y^2 + r_Z^2}\right)^2 - r_Z^2 \quad \rightarrow \quad r_{XY} = \sqrt{r_X^2 + r_Y^2}$$

Algorithm 113: h_approx
Approximate geodetic altitude from ECEF position.

Inputs:
- $[\mathbf{r}]_{\text{ecef}} \in \mathbb{R}^3$ - position resolved in ECEF frame $[\text{m}]$

Procedure:
1. Define $R_\oplus$ $[\text{m}]$, e, and f using the values from Appendix **??**.
2. Polar radius of the Earth $[\text{m}]$.

$$R_{\text{pole}} = R_\oplus (1 - f)$$

3. Extract r_X and r_Y from $[\mathbf{r}]_{\text{ecef}} = (r_X, r_Y, r_Z)^T$.
4. Scalar projection of the satellite ECEF position vector onto the XY plane of the ECEF frame $[\text{m}]$.

$$r_{XY} = \sqrt{r_X^2 + r_Y^2}$$

5. Magnitude of the ECEF position vector $[\text{m}]$.

$$r = \|[\mathbf{r}]_{\text{ecef}}\|$$

6. Approximate geodetic altitude $[\text{m}]$.

$$h \approx r - \frac{R_{\text{pole}}}{\sqrt{1 - e_\oplus^2 \left(\frac{r_{XY}}{r}\right)^2}}$$

Outputs:
- $h \in \mathbb{R}$ - approximate geodetic altitude $[\text{m}]$

10.7.2   Planecentric (Geocentric) Coordinates

The **geocentric latitude**, ϕ_{gc}, satisfies the relation

$$\sin \phi_{\text{gc}} = \frac{r_Z}{r}$$

where $[\mathbf{r}]_{\text{ecef}} = (r_X, r_Y, r_Z)^T$ is the position resolved in the ECEF frame and where $r = \|\mathbf{r}\|$. We define ϕ_{gc} to be contained in the interval $[-90°, 90°]$. Therefore, we can solve the equation for ϕ_{gc} as

$$\boxed{\phi_{\text{gc}} = \arcsin\left(\frac{r_z}{r}\right)} \tag{10.50}$$

Recall that the geodetic longitude, λ, was defined as

$$\lambda = \arctan2\left(r_Y, r_X\right)$$

Since r_X and r_Y lie in the equatorial plane, λ is an angle measured in the equatorial plane. Since the equatorial plane for both an ellipsoidal and a spherical Earth are the same, λ is the same for both. Therefore, the geocentric longitude is just equal to the geodetic longitude.

10.8   Topocentric Coordinate Frames

10.8.1   East, North, Up (ENU Frame)

Consider the ENU (local tangent) frame corresponding to a reference position, $\mathbf{r}_0$. Typically, we have $\mathbf{r}_0$ resolved in the ECEF frame. Using Algorithm 111 (`ecef2geod`), we can find the geodetic latitude and longitude, ϕ_0 and λ_0, respectively, corresponding to $[\mathbf{r}_0]_{\text{ecef}}$. The rotation matrix *from* the ECEF frame *to* the ENU frame is then given by [60, p. 37] as

$$\mathbf{R}_{\text{ecef}\to\text{enu}} = \begin{bmatrix} -\sin\lambda_0 & \cos\lambda_0 & 0 \\ -\sin\phi_0\cos\lambda_0 & -\sin\phi_0\sin\lambda_0 & \cos\phi_0 \\ \cos\phi_0\cos\lambda_0 & \cos\phi_0\sin\lambda_0 & \sin\phi_0 \end{bmatrix}$$

However, this can be written more simply as [**subirana_ecef_enu**][103, pp. 262–263]

$$\mathbf{R}_{\text{ecef}\to\text{enu}} = \mathbf{R}_1\left(\frac{\pi}{2} - \phi_0\right)\mathbf{R}_3\left(\frac{\pi}{2} + \lambda_0\right) \tag{10.51}$$

To define a formal calculation procedure for calculating $\mathbf{R}_{\text{ecef}\to\text{enu}}$, we introduce Algorithm 114.

Algorithm 114: `ecef2enu_matrix`
Rotation matrix from ECEF frame to ENU frame.

Inputs:
- $[\mathbf{r}_0]_{\text{ecef}}$ - origin of ENU frame resolved in ECEF frame

Procedure:
1. Geodetic latitude and longitude of the ENU frame's origin (Algorithm 111) $[°]$.

$$[\phi_0, \lambda_0] = \texttt{ecef2geod}([\mathbf{r}_0]_{\text{ecef}})$$

2. Convert geodetic latitude and longitude to radians (Algorithm 88).

$$\phi_0 = \texttt{deg2rad}(\phi_0)$$
$$\lambda_0 = \texttt{deg2rad}(\lambda_0)$$

3. Rotation matrix from ECEF frame to ENU frame.

$$\mathbf{R}_{\text{ecef}\to\text{enu}} = \mathbf{R}_1\left(\frac{\pi}{2} - \phi_0\right)\mathbf{R}_3\left(\frac{\pi}{2} + \lambda_0\right)$$

return • $\mathbf{R}_{\text{ecef}\to\text{enu}} \in \mathbb{R}^{3\times3}$ - rotation matrix from ECEF frame to ENU frame

---

To obtain the rotation matrix for the inverse transformation,

$$\mathbf{R}_{\text{enu}\to\text{ecef}} = \mathbf{R}_{\text{ecef}\to\text{enu}}^T = \left[\mathbf{R}_1\left(\frac{\pi}{2} - \phi_0\right)\mathbf{R}_3\left(\frac{\pi}{2} + \lambda_0\right)\right]^T = \mathbf{R}_3\left(\frac{\pi}{2} + \lambda_0\right)^T\mathbf{R}_1\left(\frac{\pi}{2} - \phi_0\right)^T$$

For an arbitrary rotation matrix $\mathbf{R}_i(\theta)$, we know $\mathbf{R}(\theta)^T = \mathbf{R}(-\theta)$. Thus,

$$\mathbf{R}_{\text{enu}\to\text{ecef}} = \mathbf{R}_3\left(-\lambda_0 - \frac{\pi}{2}\right)\mathbf{R}_1\left(\phi_0 - \frac{\pi}{2}\right)$$

> **Algorithm 115: `enu2ecef_matrix`**
> Rotation matrix from ENU frame to ECEF frame.
>
> **Inputs:**
> - $[\mathbf{r}_0]_{\text{ecef}}$ - origin of ENU frame resolved in ECEF frame
>
> **Procedure:**
>   1. Geodetic latitude and longitude of the ENU frame's origin (Algorithm 111) $[°]$.
>
> $$[\phi_0, \lambda_0] = \texttt{ecef2geod}([\mathbf{r}_0]_{\text{ecef}})$$
>
>   2. Convert geodetic latitude and longitude to radians (Algorithm 88).
>
> $$\phi_0 = \texttt{deg2rad}(\phi_0)$$
> $$\lambda_0 = \texttt{deg2rad}(\lambda_0)$$
>
>   3. Rotation matrix from ENU frame to ECEF frame.
>
> $$\mathbf{R}_{\text{enu}\to\text{ecef}} = \mathbf{R}_3\left(-\lambda_0 - \frac{\pi}{2}\right)\mathbf{R}_1\left(\phi_0 - \frac{\pi}{2}\right)$$
>
> **Outputs:**
> - $\mathbf{R}_{\text{enu}\to\text{ecef}} \in \mathbb{R}^{3\times 3}$ - rotation matrix from ENU frame to ECEF frame

Let $\mathbf{r}$ be the position vector of the satellite in a geocentric coordinate frame. Let $\boldsymbol{\rho}$ be the position vector of the satellite in the ENU frame of some reference point, where the position vector of the reference point (i.e. the origin of the ENU frame) in a geocentric coordinate frame is $\mathbf{r}_0$. Then

$$\boldsymbol{\rho} = \mathbf{r} - \mathbf{r}_0$$

Our first goal is to find $[\boldsymbol{\rho}]_{\text{enu}}$. Since we typically have $^{\text{ecef}}\mathbf{v}$ and $^{\text{ecef}}\mathbf{v}_0$ resolved in the ECEF frame,

$$\boxed{[\boldsymbol{\rho}]_{\text{ecef}} = [\mathbf{r}]_{\text{ecef}} - [\mathbf{r}_0]_{\text{ecef}}} \tag{10.52}$$

Resolving $\boldsymbol{\rho}$ in the ENU frame,

$$\boxed{[\boldsymbol{\rho}]_{\text{enu}} = \mathbf{R}_{\text{ecef}\to\text{enu}}[\boldsymbol{\rho}]_{\text{ecef}}} \tag{10.53}$$

Next, let $^{\text{ecef}}\mathbf{v}$ and $^{\text{ecef}}\mathbf{v}_0$ be the ECEF velocities of the satellite and the reference point (i.e. the origin of the ENU frame). Then

$$^{\text{ecef}}\dot{\boldsymbol{\rho}} = {}^{\text{ecef}}\mathbf{v} - {}^{\text{ecef}}\mathbf{v}_0$$

Since we typically have $^{\text{ecef}}\mathbf{v}$ and $^{\text{ecef}}\mathbf{v}_0$ resolved in the ECEF frame,

$$\boxed{\left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{ecef}} = \left[^{\text{ecef}}\mathbf{v}\right]_{\text{ecef}} - \left[^{\text{ecef}}\mathbf{v}_0\right]_{\text{ecef}}} \tag{10.54}$$

Let $\dot{\boldsymbol{\rho}}$ be the satellite velocity relative to the ENU frame (i.e. $\dot{\boldsymbol{\rho}} = {}^{\text{enu}}\dot{\boldsymbol{\rho}}$). Then

$$\dot{\boldsymbol{\rho}} = {}^{\text{ecef}}\dot{\boldsymbol{\rho}} + \boldsymbol{\omega}_{\text{enu}/\text{ecef}} \times \boldsymbol{\rho}$$

Since we have $\boldsymbol{\rho}$ and $^{\text{ecef}}\dot{\boldsymbol{\rho}}$ resolved in the ECEF frame as $[\boldsymbol{\rho}]_{\text{ecef}}$ and $\left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{ecef}}$, we can calculate

$$\boxed{[\dot{\boldsymbol{\rho}}]_{\text{ecef}} = \left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{ecef}} + \left[\boldsymbol{\omega}_{\text{enu}/\text{ecef}}\right]_{\text{ecef}} \times [\boldsymbol{\rho}]_{\text{ecef}}} \tag{10.55}$$

where $\left[\boldsymbol{\omega}_{\text{enu}/\text{ecef}}\right]_{\text{ecef}}$ can be found using Algorithm 117 with the inputs $[\mathbf{r}_0]_{\text{ecef}}$ and $\left[^{\text{ecef}}\mathbf{v}_0\right]_{\text{ecef}}$ (see Section **??**).
Finally, resolving $\dot{\boldsymbol{\rho}}$ in the ENU frame,

$$\boxed{[\dot{\boldsymbol{\rho}}]_{\text{enu}} = \mathbf{R}_{\text{ecef}\to\text{enu}}[\dot{\boldsymbol{\rho}}]_{\text{ecef}}} \tag{10.56}$$

Algorithm 116: `ecef2enu`
ENU position and velocity from ECEF position and velocity.

Inputs:
- $[\mathbf{r}]_{\text{ecef}}$ - satellite position resolved in ECEF frame
- $[\mathbf{r}_0]_{\text{ecef}}$ - origin of ENU frame resolved in ECEF frame
- $\left[^{\text{ecef}}\mathbf{v}\right]_{\text{ecef}}$ - *(OPTIONAL)* satellite ECEF velocity resolved in ECEF frame
- $\left[^{\text{ecef}}\mathbf{v}_0\right]_{\text{ecef}}$ - *(OPTIONAL)* ECEF velocity of ENU frame's origin resolved in ECEF frame

Procedure:
1. Rotation matrix from ECEF frame to ENU frame (Algorithm 114).

$$\mathbf{R}_{\text{ecef}\to\text{enu}} = \texttt{ecef2enu\_matrix}([\mathbf{r}_0]_{\text{ecef}})$$

2. ENU position resolved in ECEF frame.

$$[\boldsymbol{\rho}]_{\text{ecef}} = [\mathbf{r}]_{\text{ecef}} - [\mathbf{r}_0]_{\text{ecef}}$$

3. ENU position resolved in ENU frame.

$$[\boldsymbol{\rho}]_{\text{enu}} = \mathbf{R}_{\text{ecef}\to\text{enu}}[\boldsymbol{\rho}]_{\text{ecef}}$$

4. Determine ENU velocity if the satellite velocity is provided.

if $\left[^{\text{ecef}}\mathbf{v}\right]_{\text{ecef}}$ is input
     (a) Default $\left[^{\text{ecef}}\mathbf{v}_0\right]_{\text{ecef}}$ to $\mathbf{0}$ if not input (assume reference point is stationary in ECEF frame unless otherwise specified).
     (b) ECEF relative velocity resolved in ECEF frame.

$$\left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{ecef}} = \left[^{\text{ecef}}\mathbf{v}\right]_{\text{ecef}} - \left[^{\text{ecef}}\mathbf{v}_0\right]_{\text{ecef}}$$

     (c) Angular velocity of ENU frame relative to *and* resolved in ECEF frame (Algorithm **??**) $[\text{rad/s}]$.

$$\left[\boldsymbol{\omega}_{\text{enu/ecef}}\right]_{\text{ecef}} = \texttt{rv2w}\left([\mathbf{r}_0]_{\text{ecef}}, \left[^{\text{ecef}}\mathbf{v}_0\right]_{\text{ecef}}\right)$$

     (d) ENU velocity resolved in ECEF frame.

$$[\dot{\boldsymbol{\rho}}]_{\text{ecef}} = \left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{ecef}} + \left[\boldsymbol{\omega}_{\text{enu/ecef}}\right]_{\text{ecef}} \times [\boldsymbol{\rho}]_{\text{ecef}}$$

     (e) ENU velocity resolved in ENU frame.

$$[\dot{\boldsymbol{\rho}}]_{\text{enu}} = \mathbf{R}_{\text{ecef}\to\text{enu}}[\dot{\boldsymbol{\rho}}]_{\text{ecef}}$$

end

Outputs:
- $[\boldsymbol{\rho}]_{\text{enu}} \in \mathbb{R}^3$ - satellite position resolved in ENU frame
- $[\dot{\boldsymbol{\rho}}]_{\text{enu}} \in \mathbb{R}^3$ - satellite ENU velocity resolved in ENU frame

We can also define Algorithm **??** by simply inverting all the steps in Algorithm 117.

Algorithm 117: `enu2ecef`
ECEF position and velocity from ENU position and velocity.

Inputs:
- $[\boldsymbol{\rho}]_{\text{enu}}$ - satellite position resolved in ENU frame
- $[\mathbf{r}_0]_{\text{ecef}}$ - origin of ENU frame resolved in ECEF frame
- $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ - *(OPTIONAL)* satellite ENU velocity resolved in ENU frame
- $[^{\text{ecef}}\mathbf{v}_0]_{\text{ecef}}$ - *(OPTIONAL)* ECEF velocity of ENU frame's origin resolved in ECEF frame

Procedure:

1. Rotation matrix from ENU frame to ECEF frame (Algorithm 115).

$$\mathbf{R}_{\text{enu}\to\text{ecef}} = \texttt{enu2ecef\_matrix}([\mathbf{r}_0]_{\text{ecef}})$$

2. ENU position resolved in ECEF frame.

$$[\boldsymbol{\rho}]_{\text{ecef}} = \mathbf{R}_{\text{enu}\to\text{ecef}}[\boldsymbol{\rho}]_{\text{enu}}$$

3. ECEF position resolved in ECEF frame.

$$[\mathbf{r}]_{\text{ecef}} = [\mathbf{r}_0]_{\text{ecef}} + [\boldsymbol{\rho}]_{\text{ecef}}$$

4. Determine ECEF velocity if the satellite velocity is provided.

if $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ is input

   (a) Default $[^{\text{ecef}}\mathbf{v}_0]_{\text{ecef}}$ to $\mathbf{0}$ if not input (assume reference point is stationary in ECEF frame unless otherwise specified).

   (b) Angular velocity of ENU frame relative to *and* resolved in ECEF frame (Algorithm **??**).

   $$\left[\boldsymbol{\omega}_{\text{enu/ecef}}\right]_{\text{ecef}} = \texttt{rv2w}\left([\mathbf{r}_0]_{\text{ecef}}, [^{\text{ecef}}\mathbf{v}_0]_{\text{ecef}}\right)$$

   (c) Angular velocity of ECEF frame relative to *and* resolved in ENU frame [rad/s].

   $$\left[\boldsymbol{\omega}_{\text{ecef/enu}}\right]_{\text{enu}} = -\mathbf{R}_{\text{enu}\to\text{ecef}}^{T}\left[\boldsymbol{\omega}_{\text{enu/ecef}}\right]_{\text{ecef}}$$

   (d) ECEF relative velocity resolved in ENU frame.

   $$\left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{enu}} = [\dot{\boldsymbol{\rho}}]_{\text{enu}} + \left[\boldsymbol{\omega}_{\text{ecef/enu}}\right]_{\text{enu}} \times [\boldsymbol{\rho}]_{\text{enu}}$$

   (e) ECEF relative velocity resolved in ECEF frame.

   $$\left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{ecef}} = \mathbf{R}_{\text{enu}\to\text{ecef}}\left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{enu}}$$

   (f) ECEF velocity resolved in ECEF frame.

   $$\left[^{\text{ecef}}\mathbf{v}\right]_{\text{ecef}} = \left[^{\text{ecef}}\mathbf{v}_0\right]_{\text{ecef}} + \left[^{\text{ecef}}\dot{\boldsymbol{\rho}}\right]_{\text{ecef}}$$

end

Outputs:
- $[\mathbf{r}]_{\text{ecef}}$ - satellite position resolved in ECEF frame
- $[^{\text{ecef}}\mathbf{v}]_{\text{ecef}}$ - satellite ECEF velocity resolved in ECEF frame

10.8.2   North, East, Down (NED) Frame

10.9   Azimuth, Elevation, Slant Range and Their Rates

Consider the position, $\boldsymbol{\rho}$, and velocity, $\dot{\boldsymbol{\rho}}$, of a satellite relative to the origin of the ENU frame (a local tangent coordinate frame), as discussed in Section **??**. The satellite's **range** (i.e. distance from origin of ENU frame) is defined as

$$\boxed{\rho = \|\boldsymbol{\rho}\|} \tag{10.57}$$

The satellite's **azimuth** angle is defined as the angle measured clockwise from the N (north) axis towards the E (east) axis about the U (up) axis. It satisfies the relation

$$\tan\left(\text{Az}\right) = \frac{\rho_E}{\rho_N} \tag{10.58}$$

The satellite's **elevation** angle is defined as the angle measured from the EN plane to the satellite's ENU position vector, $\boldsymbol{\rho}$ [**damico_279a_lecture_8**][**tapley_statistical_orbit_determination**]. It satisfies the relation [60, p. 37]

$$\tan\left(\text{El}\right) = \frac{\rho_U}{\sqrt{\rho_E^2 + \rho_N^2}} \tag{10.59}$$

Let's define $[\boldsymbol{\rho}]_{\text{enu}}$ in terms of its components as

$$[\boldsymbol{\rho}]_{\text{enu}} = \begin{bmatrix} \rho_E \\ \rho_N \\ \rho_U \end{bmatrix}$$

We define the azimuth, Az, to be contained in the interval $[0, 360°)$, and the elevation, El, to be contained in the interval $[-90°, 90°]$. To calculate the azimuth, we have to use the four-quadrant inverse tangent to ensure that Az is returned in the correct quadrant. Note that arctan2 is typically programmed such that it returns a value in the interval $[-180°, 180°)$. Therefore, in its implementation, we must wrap the result of Eq. (10.60) to the interval $[0, 360°)$, which can be done using the modulo operator.

$$\boxed{\text{Az} = \text{arctan2}\left(\rho_E, \rho_N\right)} \tag{10.60}$$

Since the inverse tangent function already returns values in the interval $[-90°, 90°]$, the elevation can be computed simply as

$$\boxed{\text{El} = \arctan\left(\frac{\rho_U}{\sqrt{\rho_E^2 + \rho_N^2}}\right)} \tag{10.61}$$

We can also define the range, azimuth, and elevation *rates*. The **range rate** is simply

$$\boxed{\dot{\rho} = \|\dot{\boldsymbol{\rho}}\|} \tag{10.62}$$

Let's express $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ in terms of its components as

$$[\dot{\boldsymbol{\rho}}]_{\text{enu}} = \begin{bmatrix} \dot{\rho}_E \\ \dot{\rho}_N \\ \dot{\rho}_U \end{bmatrix}$$

To find the **elevation rate**, recall Eq. (10.66):

$$\begin{bmatrix} \dot{\rho}_E \\ \dot{\rho}_N \\ \dot{\rho}_U \end{bmatrix} = \begin{bmatrix} \dot{\rho}\cos\left(\text{El}\right)\sin\left(\text{Az}\right) + \rho\cos\left(\text{Az}\right)\cos\left(\text{El}\right)\dot{\text{A}}\text{z} - \rho\sin\left(\text{El}\right)\sin\left(\text{Az}\right)\dot{\text{E}}\text{l} \\ \dot{\rho}\cos\left(\text{El}\right)\cos\left(\text{Az}\right) - \rho\sin\left(\text{Az}\right)\cos\left(\text{El}\right)\dot{\text{A}}\text{z} - \rho\sin\left(\text{El}\right)\cos\left(\text{Az}\right)\dot{\text{E}}\text{l} \\ \dot{\rho}\sin\left(\text{El}\right) + \rho\cos\left(\text{El}\right)\dot{\text{E}}\text{l} \end{bmatrix}$$

From the last row of this matrix equation, we have

$$\dot{\rho}_U = \dot{\rho}\sin\left(\text{El}\right) + \rho\cos\left(\text{El}\right)\dot{\text{E}}\text{l}$$

Solving for $\dot{\text{El}}$,

$$\dot{\text{El}} = \frac{\dot{\rho}_U - \dot{\rho}\sin{(\text{El})}}{\rho\cos{(\text{El})}} \tag{10.63}$$

To find the **azimuth rate**, $\dot{\text{Az}}$, let's differentiate both sides of Eq. (10.58).

$$\frac{d}{dt}\tan{(\text{Az})} = \frac{d}{dt}\left(\frac{\rho_E}{\rho_N}\right) \quad \rightarrow \quad \sec^2{(\text{Az})}\dot{\text{Az}} = \frac{\rho_N\dot{\rho}_E - \rho_E\dot{\rho}_N}{\rho_N^2}$$

Solving for $\dot{\text{Az}}$, noting that $\sec^2{(\cdot)} = 1/\cos^2{(\cdot)}$,

$$\dot{\text{Az}} = \left(\frac{\rho_N\dot{\rho}_E - \rho_E\dot{\rho}_N}{\rho_N^2}\right)\cos^2{(\text{Az})} \tag{10.64}$$

Algorithm 118: `enu2aer`
Range, azimuth, elevation, and their rates from ENU position and velocity.

Inputs:
- $[\boldsymbol{\rho}]_{\text{enu}} \in \mathbb{R}^3$ - position resolved in ENU frame [any unit]
- $[\dot{\boldsymbol{\rho}}]_{\text{enu}} \in \mathbb{R}^3$ - *(OPTIONAL)* ENU velocity resolved in ENU frame [any unit]

Procedure:
1. Extract ρ_E, ρ_N, and ρ_U from $[\boldsymbol{\rho}]_{\text{enu}}$.
2. Azimuth [°].

$$\text{Az} = \text{mod}\left[\arctan2{(\rho_E, \rho_N)}, 360°\right]$$

3. Elevation [°].

$$\text{El} = \arctan\left(\frac{\rho_U}{\sqrt{\rho_E^2 + \rho_N^2}}\right)$$

4. Range.

$$\rho = \|[\boldsymbol{\rho}]_{\text{enu}}\|$$

5. Extract $\dot{\rho}_E$, $\dot{\rho}_N$, and $\dot{\rho}_U$ from $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ (only do this if $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ is specified).
6. Range rate (only calculate if $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ is specified).

$$\dot{\rho} = \|[\dot{\boldsymbol{\rho}}]_{\text{enu}}\|$$

7. Azimuth rate (only calculate if $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ is specified) [°/s].

$$\dot{\text{Az}} = \left(\frac{\rho_N\dot{\rho}_E - \rho_E\dot{\rho}_N}{\rho_N^2}\right)\cos^2{(\text{Az})}$$

8. Elevation rate (only calculate if $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ is specified) [°/s].

$$\dot{\text{El}} = \frac{\dot{\rho}_U - \dot{\rho}\sin{(\text{El})}}{\rho\cos{(\text{El})}}$$

Outputs:
- $\text{Az} \in \mathbb{R}$ - azimuth [°]
- $\text{El} \in \mathbb{R}$ - elevation [°]
- $\rho \in \mathbb{R}$ - range [any unit]
- $\dot{\text{Az}} \in \mathbb{R}$ - azimuth rate [°/s]
- $\dot{\text{El}} \in \mathbb{R}$ - elevation rate [°/s]
- $\dot{\rho} \in \mathbb{R}$ - range rate [any unit]

Note:
- $[\boldsymbol{\rho}]_{\text{enu}}$ and $[\dot{\boldsymbol{\rho}}]_{\text{enu}}$ can be input in any units, but they *must* be consistent if both are input.
- Az is measured clockwise from N (north) axis towards E (east) axis about the U (up) axis.
- El is measured from the EN plane to the satellite's ENU position vector, $\boldsymbol{\rho}$.
- $-90° \leq \text{El} \leq 90°$
- $0° \leq \text{Az} < 360°$

To perform the conversion in the opposite direction, we can first compute the satellite's ENU position simply as [**damico_279a_lecture_8**]

$$[\boldsymbol{\rho}]_{\text{enu}} = \rho \begin{bmatrix} \cos{(\text{El})}\sin{(\text{Az})} \\ \cos{(\text{El})}\cos{(\text{Az})} \\ \sin{(\text{El})} \end{bmatrix} \qquad (10.65)$$

To find its ENU velocity, we can differentiate Eq. (10.65)

$$[\dot{\boldsymbol{\rho}}]_{\text{enu}} = \rho \frac{d}{dt} \begin{bmatrix} \cos{(\text{El})}\sin{(\text{Az})} \\ \cos{(\text{El})}\cos{(\text{Az})} \\ \sin{(\text{El})} \end{bmatrix} + \frac{d\rho}{dt} \begin{bmatrix} \cos{(\text{El})}\sin{(\text{Az})} \\ \cos{(\text{El})}\cos{(\text{Az})} \\ \sin{(\text{El})} \end{bmatrix}$$

$$= \rho \begin{bmatrix} \cos{(\text{Az})}\cos{(\text{El})}\dot{\text{Az}} - \sin{(\text{El})}\sin{(\text{Az})}\dot{\text{El}} \\ -\sin{(\text{Az})}\cos{(\text{El})}\dot{\text{Az}} - \sin{(\text{El})}\cos{(\text{Az})}\dot{\text{El}} \\ \cos{(\text{El})}\dot{\text{El}} \end{bmatrix} + \dot{\rho} \begin{bmatrix} \cos{(\text{El})}\sin{(\text{Az})} \\ \cos{(\text{El})}\cos{(\text{Az})} \\ \sin{(\text{El})} \end{bmatrix}$$

$$[\dot{\boldsymbol{\rho}}]_{\text{enu}} = \begin{bmatrix} \dot{\rho}\cos{(\text{El})}\sin{(\text{Az})} + \rho\cos{(\text{Az})}\cos{(\text{El})}\dot{\text{Az}} - \rho\sin{(\text{El})}\sin{(\text{Az})}\dot{\text{El}} \\ \dot{\rho}\cos{(\text{El})}\cos{(\text{Az})} - \rho\sin{(\text{Az})}\cos{(\text{El})}\dot{\text{Az}} - \rho\sin{(\text{El})}\cos{(\text{Az})}\dot{\text{El}} \\ \dot{\rho}\sin{(\text{El})} + \rho\cos{(\text{El})}\dot{\text{El}} \end{bmatrix} \qquad (10.66)$$

Algorithm 119: `aer2enu`

ENU position and velocity from range, azimuth, elevation, and their rates.

Inputs:
- $\text{Az} \in \mathbb{R}$ - azimuth [°]
- $\text{El} \in \mathbb{R}$ - elevation [°]
- $\rho \in \mathbb{R}$ - range [any unit]
- $\dot{\text{Az}} \in \mathbb{R}$ - *(OPTIONAL)* azimuth rate [°/s]
- $\dot{\text{El}} \in \mathbb{R}$ - *(OPTIONAL)* elevation rate [°/s]
- $\dot{\rho} \in \mathbb{R}$ - *(OPTIONAL)* range rate [any unit]

Procedure:
1. Position resolved in ENU frame.

$$[\boldsymbol{\rho}]_{\text{enu}} = \begin{bmatrix} \cos{(\text{El})}\sin{(\text{Az})} \\ \cos{(\text{El})}\cos{(\text{Az})} \\ \sin{(\text{El})} \end{bmatrix}$$

2. ENU velocity resolved in ENU frame (only calculate if $\dot{\rho}$, $\dot{\text{Az}}$, and $\dot{\text{El}}$ are specified).

$$[\dot{\boldsymbol{\rho}}]_{\text{enu}} = \begin{bmatrix} \dot{\rho}\cos{(\text{El})}\sin{(\text{Az})} + \rho\cos{(\text{Az})}\cos{(\text{El})}\dot{\text{Az}} - \rho\sin{(\text{El})}\sin{(\text{Az})}\dot{\text{El}} \\ \dot{\rho}\cos{(\text{El})}\cos{(\text{Az})} - \rho\sin{(\text{Az})}\cos{(\text{El})}\dot{\text{Az}} - \rho\sin{(\text{El})}\cos{(\text{Az})}\dot{\text{El}} \\ \dot{\rho}\sin{(\text{El})} + \rho\cos{(\text{El})}\dot{\text{El}} \end{bmatrix}$$

return
- $[\boldsymbol{\rho}]_{\mathrm{enu}} \in \mathbb{R}^3$ - position resolved in ENU frame [same units as ρ]
- $[\dot{\boldsymbol{\rho}}]_{\mathrm{enu}} \in \mathbb{R}^3$ - ENU velocity resolved in ENU frame [same units as $\dot{\rho}$]

Note:
- ρ and $\dot{\rho}$ can be input in any units, but they *must* be consistent if both are input.
- Az is measured clockwise from N (north) axis towards E (east) axis about the U (up) axis.
- El is measured from the EN plane to the satellite's ENU position vector, $\boldsymbol{\rho}$.
- $-90° \leq \mathrm{El} \leq 90°$
- $0° \leq \mathrm{Az} < 360°$

10.10   Geocentric Right Ascension and Declination

The geocentric right ascension (α) and declination (δ) are defined as shown in Fig. 10.2. Directly from Fig. 10.2, we
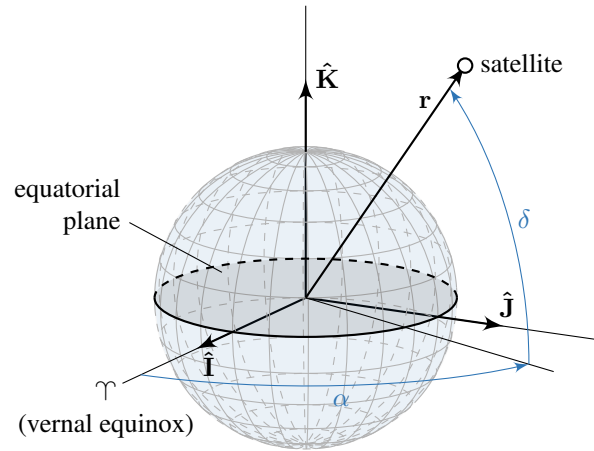
Figure 10.2: Geocentric right ascension and declination (adapted from Figure 4-5 in [103, p. 257].

can write the position of the satellite, resolved in the ECI frame, as

$$[\mathbf{r}]_{\mathrm{eci}} = \begin{bmatrix} r_I \\ r_J \\ r_K \end{bmatrix} = r \begin{bmatrix} \cos\delta\cos\alpha \\ \cos\delta\sin\alpha \\ \sin\alpha \end{bmatrix}$$

Since α is the angle measured from the I-component of $[\mathbf{r}]_{\mathrm{eci}}$ to the J-component of $[\mathbf{r}]_{\mathrm{eci}}$, we have

$$\tan\alpha = \frac{r_J}{r_I} \quad \rightarrow \quad \boxed{\alpha = \mathrm{arctan2}\,(r_J, r_I)} \tag{10.67}$$

δ is the angle between $[\mathbf{r}]_{\mathrm{eci}}$ and the projection of $[\mathbf{r}]_{\mathrm{eci}}$ onto the IJ plane. Since this projection is given by $\sqrt{r_I^2 + r_J^2}$, we have [103, pp. 256–259][60, p. 25].

$$\tan\delta = \frac{r_K}{\sqrt{r_I^2 + r_J^2}} \quad \rightarrow \quad \boxed{\delta = \mathrm{arctan2}\left(r_K, \sqrt{r_I^2 + r_J^2}\right)} \tag{10.68}$$

Thus, we can develop a simple algorithm to compute the geocentric right ascension and declination of a satellite given its position resolved in the ECI frame.

Algorithm 120: `r2ad`
ECI position to geocentric right ascension and declination.

Inputs:

- $[\mathbf{r}]_{\mathrm{eci}} \in \mathbb{R}^3$ - position resolved in ECI frame [any unit]

Procedure:

1. Extract r_I, r_J, and r_K from $[\mathbf{r}]_{\mathrm{eci}}$.
2. Right ascension.

$$\alpha = \arctan2\left(r_J, r_I\right)$$

3. Declination.

$$\delta = \arctan2\left(r_K, \sqrt{r_I^2 + r_J^2}\right)$$

Outputs:

- $\alpha \in \mathbb{R}$ - right ascension [rad]
- $\delta \in \mathbb{R}$ - declination [rad]

11

Gravity

TODO: update all tikz diagrams in separate latex doc TODO: note on outside of body TODO: return statements TODO: test cases TODO: rewrite in terms of physical vectors TODO: generalize (not specific to Earth) TODO: test cases in `https://ntrs.nasa.gov/api/citations/19940025085/downloads/19940025085.pdf` TODO: geopotential

11.1 Spherical Harmonic Gravitational Potential

The (conventional) Legendre polynomial of degree n is defined as [103, p. 540]

$$P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} (x^2 - 1)^n = \frac{1}{2^n} \sum_{j=0}^{n/2} \left[\left(\frac{(-1)^j (2n - 2j)!}{j!(n-j)!(n-2j)!} \right) x^{n-2j} \right] \tag{11.1}$$

The associated Legendre polynomial of degree n and order m is defined as [103, p. 543]

$$P_{nm}(x) = (1 - x^2)^{n/2} \frac{d^m}{dx^m} P_n(x) \tag{11.2}$$

The gravitational potential of a planetary body can be defined in terms of these polynomials as a function of radial distance (r), planetocentric latitude (ϕ_{pc}), and planetocentric longitude (λ_{pc}) [60, p. 57], [103, p. 545].

$$U(r, \phi_{\mathrm{pc}}, \lambda_{\mathrm{pc}}) = \frac{\mu}{r} \sum_{n=0}^{\infty} \sum_{m=0}^{n} \left[\left(\frac{R}{r} \right)^n P_{nm}(\sin \phi_{\mathrm{pc}}) (C_{nm} \cos{(m\lambda_{\mathrm{pc}})} + S_{nm} \sin{(m\lambda_{\mathrm{pc}})}) \right] \tag{11.3}$$

In the equation[1] above, μ is the **standard gravitational parameter** of the body, R is the equatorial radius[2] of the

---

[1] The derivation of this equation is quite involved. Some aspects of the derivation are provided in [**geopotential_model_wikipedia**], along with a compact derivation of spherical harmonics. Another simplified derivation is provided by [60, pp. 56–57], while a more detailed derivation is provided by [103, pp. 538–545].

[2] Many references, such as [26] and [68, p. 79] write the gravitational potential in terms of a (or a_e), which is typically used to denote the semi-major axis (i.e. equatorial radius) of a reference ellipsoid. Here, we use the symbol R to denote the equatorial radius, since the symbol a will be used to denote some auxiliary parameters used in the evaluation of the gravitational acceleration and its Jacobian later. Note that [103, p. 545] and [60, p. 57] use $R_\oplus$ in their geopotential equations, since they wrote them in the context of the geopotential of the Earth (which is denoted with the astronomical symbol $\oplus$).

body, and C_{nm} and S_{nm} are **gravitational coefficients**. These coefficients are also grouped into the following three subclasses [60, pp. 57–58], [103, pp. 547–549]:

- **zonal coefficients:** $m = 0$ (form bands of latitude, since they describe the part of the potential that does not depend on longitude)
- **sectorial coefficients:** $m = n$ (form bands of longitude, since they describe the part of the potential that does not depend on latitude)
- **tesseral coefficients:** $m < n$ (describe a "tile" of the planetary body)

   TODO However, we can note from this equation that

$$\boxed{m \leq n} \tag{11.4}$$

> **Note:** The gravitational potential of the Earth is referred to as the **geopotential** of the Earth.

11.1.1 J_2 and the Zonal Harmonic Coefficients

Recall that the zonal potential coefficients all have order 0. Thus, the **zonal potential coefficient of degree** n is denoted $C_{n,0}$. The **nth zonal harmonic coefficient** (or the **zonal harmonic coefficient of degree n**) is defined as the *negative* of the corresponding zonal potential coefficient [60, p. 58], [103, p. 545].

$$\boxed{J_n = -C_{n,0}} \tag{11.5}$$

The most commonly used zonal harmonic is the **2nd zonal harmonic coefficient**, J_2. For planetary bodies, this term represents the flattening of the body (or equivalently their equatorial bulge) that forms due to the rotation about their 3rd axis.

$$\boxed{J_2 = -C_{2,0}} \tag{11.6}$$

11.1.2   Other Special Gravitational Coefficients

Gravitational potential models are defined such that the center of the coordinate system is located at the body's center of mass. This results in $C_{1,0}$, $C_{1,1}$, and $S_{1,1}$ all being identically zero [103, p. 545], [60, p. 61], [30, p. 7]. By definition, we also have that $S_{1,0}$ is zero [103, p. 545] and that $C_{0,0}$ is one [60, p. 59]. Additionally, $S_{0,0}$ is zero by definition. We summarize these additional six special gravitational coefficients in Table 11.1 below.

n	m	C_{nm}	S_{nm}
0	0	1	0
1	0	0	0
1	1	0	0

Table 11.1: Special gravitational coefficients.

11.1.3   Normalized Gravitational Coefficients and Associated Legendre Polynomials

The C_{nm} and S_{nm} gravitational coefficients are typically given in normalized form as the **normalized gravitational coefficients** $\bar{C}_{nm}$ and $\bar{S}_{nm}$, respectively. The normalizations are defined by

$$\boxed{\bar{C}_{nm} = \frac{C_{nm}}{N_{nm}}} \tag{11.7}$$

$$\boxed{\bar{S}_{nm} = \frac{S_{nm}}{N_{nm}}}$$

(11.8)

where the **Kaula normalization factor** (or simply the **normalization factor**), N_{nm}, is given by

$$\boxed{N_{nm} = \sqrt{\frac{(n-m)!(2n+1)(2-\delta_{0,m})}{(n+m)!}}}$$

(11.9)

and where $\delta_{i,j}$ is the Kronecker delta function[3], defined as

$$\boxed{\delta_{i,j} = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases}}$$

(11.10)

The normalized associated Legendre polynomial is defined as

$$\boxed{\bar{P}_{nm} = P_{nm} N_{nm}}$$

(11.11)

TODO: reference (Eckman)

11.1.4 Defining Parameters of a Gravity Model

Gravity models are defined using the following TODO add more description:

Parameter	Symbol	Units
standard gravitational parameter	μ	$\mathrm{kg/m^3}$
mean equatorial radius (i.e. ellipsoid semi-major axis)	R	m
normalized gravitational coefficients (tabulated)	$\bar{C}_{nm}, \bar{S}_{nm}$	-
maximum possible degree	$N_{\max}$	-
maximum possible order	$M_{\max}$	-
tide system TODO list types	-	-
Love number TODO	TODO	-
num/eq for converting between tide systems	TODO	-
coordinate frame	-	-

Table 11.2: Defining parameters of a gravity model.

11.1.5 Square Gravity Models

Recall Eq. (11.3), repeated below for convenience, defining a spherical harmonic gravitational potential.

$$U(r, \phi_{\mathrm{pc}}, \lambda_{\mathrm{pc}}) = \frac{\mu}{r} \sum_{n=0}^{\infty} \sum_{m=0}^{n} \left[\left(\frac{R}{r}\right)^n P_{nm}(\sin \phi_{\mathrm{pc}})(C_{nm} \cos{(m\lambda_{\mathrm{pc}})} + S_{nm} \sin{(m\lambda_{\mathrm{pc}})}) \right]$$

(11.3)

This equation expands the summation out to infinity. However, we do not have an infinite number of the C_{nm} and S_{nm} coefficients. Therefore, we define

$$\boxed{N = \text{maximum degree/order of the gravity model}}$$

---

[3] TODO Other languages In MATLAB, we can use `eq(i,j)` for the Kronecker delta function.

and write the gravitational potential as

$$U(r, \phi_{\mathrm{pc}}, \lambda_{\mathrm{pc}}) = \frac{\mu}{r} \sum_{n=0}^{N} \sum_{m=0}^{n} \left[\left(\frac{R}{r}\right)^n P_{nm}(\sin \phi_{\mathrm{pc}})(C_{nm} \cos{(m\lambda_{\mathrm{pc}})} + S_{nm} \sin{(m\lambda_{\mathrm{pc}})}) \right]$$

Let's rewrite this potential is written in a slightly different form. First, note that $(R/r)^n$ is independent of m, so we can pull it out from under the second summation.

$$U(r, \phi_{\mathrm{pc}}, \lambda_{\mathrm{pc}}) = \frac{\mu}{r} \sum_{n=0}^{N} \left(\frac{R}{r}\right)^n \sum_{m=0}^{n} P_{nm}(\sin \phi_{\mathrm{pc}})(C_{nm} \cos{(m\lambda_{\mathrm{pc}})} + S_{nm} \sin{(m\lambda_{\mathrm{pc}})})$$

For $n = m = 0$, we can recall from Section 11.1.2 that $C_{0,0} = 1$ and $S_{1,1} = 0$. Thus, we can write the gravitational potential as

$$U(r, \phi_{\mathrm{pc}}, \lambda_{\mathrm{pc}}) = \frac{\mu}{r} \left[1 + \sum_{n=1}^{N} \left(\frac{R}{r}\right)^n \sum_{m=0}^{n} P_{nm}(\sin \phi_{\mathrm{pc}})(C_{nm} \cos{(m\lambda_{\mathrm{pc}})} + S_{nm} \sin{(m\lambda_{\mathrm{pc}})}) \right]$$

Also in Section 11.1.2, we noted that $C_{1,0} = C_{1,1} = S_{1,1} = 0$. Thus, we can start the outer summation from $n = 2$ [30, p. 7].

$$U(r, \phi_{\mathrm{pc}}, \lambda_{\mathrm{pc}}) = \frac{\mu}{r} \left[1 + \sum_{n=2}^{N} \left(\frac{R}{r}\right)^n \sum_{m=0}^{n} P_{nm}(\sin \phi_{\mathrm{pc}})(C_{nm} \cos{(m\lambda_{\mathrm{pc}})} + S_{nm} \sin{(m\lambda_{\mathrm{pc}})}) \right] \qquad (11.12)$$

TODO: normalized form

11.1.6 Tidal Effects

Eq. (11.3) describes the gravitational potential of the Earth in terms of the C_{nm} and S_{nm} gravitational coefficients. Strictly speaking, this coefficients are ***not constant*** due to **tidal effects**. The gravitational forces exerted by neighboring bodies on a planetary body will cause both its solid and liquid components to deform. For example, the gravity of the Moon not only causes our ocean tides (by deforming the Earth's oceans), but it also deforms the Earth's solid body. These deformations, which we refer to as **solid tides** and **ocean tides**, cause both permanent and periodic changes in a body's gravitational field.

There are many different definitions and conventions when considering tidal effects, and it often becomes very confusing deciphering what each actually means. Here, we lay them out in a sequential order [68, pp. 15–17], [53, p. 2]:

1. **Tide-Free Gravitational Potential:** The **true** gravitational potential that a body would have in free space, without the influence of external bodies. *Note that the tide-free gravitational potential is an idealization that is not physically observable.*
2. **Conventional Tide-Free Gravitational Potential:** The *modeled* gravitational potential that a body would have in free space, without the influence of external bodies.
3. **Zero-Tide Gravitational Potential:** The gravitational forces produced by external bodies will cause the body we are considering to physically deform. This physical deformation results in the body having a different gravitational potential. One component of the deformation is permanent, while the other is periodic. The zero-tide potential is the gravitational potential that the body would produce considering only its permanent deformations from external forces.
4. **Mean-Tide Gravitational Potential:** The external bodies producing the tidal effects have gravitational potentials. We refer to this potential as the **tide-generating potential**. These tide-generating potential also have a permanent (time-averaged) component and a periodic component. The mean-tide gravitational potential is the zero-tide gravitational potential with the addition of the permanent component of the tide-generating potential. Thus, the mean-tide gravitational potential also incorporates a portion of the gravitational field produced by external bodies (masses); it is not generated by the body in question alone.

11.1.7   Tide-Free vs. Zero-Tide Gravitational Potential Models

Recall from Section 11.1.4 that spherical harmonic gravity models of bodies in our solar system are defined using the normalized gravitational coefficients, $\bar{C}_{nm}$ and $\bar{S}_{nm}$. Gravity models define these coefficients as constants in a tabulated form. Immediately, this means that these coefficients either correspond to a mean-tide, zero-tide, or conventional tide-free gravitational potential model.

Resolution 16 of the 18th General Assembly of the IAG (1984) recommended that zero-tide values be used to define geopotential models. However, in practice, models are defined using both zero-tide or conventional tide-free values, depending on the model[4].

TODO The tidal effects are applied as corrections to the gravitational coefficients.

$$(C_{nm})_{\text{corrected}} = C_{nm} + \Delta C_{nm}$$
$$(S_{nm})_{\text{corrected}} = S_{nm} + \Delta S_{nm}$$

The corrections ΔC_{nm} and ΔS_{nm} are themselves modeled using a spherical harmonic model in a similar way to that of the static gravitational potential. However, their computation becomes substantially more complicated than the static gravitational potential. Additionally, the inclusion of their effects is minimal; as an example, [68, p. 92] cites two orbits where the tidal effects amounted to a position error on the order of millimeters over the course of about a week.

11.2   Coefficient Tables and Vectors

TODO: unnormalized coefficient vectors The C_{nm} and S_{nm} coefficients are typically given in a normalized form as $\bar{C}_{nm}$ and $\bar{S}_{nm}$, respectively (see Section 11.1.3). The data files storing the coefficients may have various extensions, but these files can generally be opened as simple text (`.txt`) files. The files themselves have the data in a tabular form. While there may be additional columns (typically related to the uncertainties corresponding to the coefficients), the tables always store at the very least the following columns:

n	m	$\bar{C}_{nm}$	$\bar{S}_{nm}$
0	0	$\bar{C}_{0,0}$	$\bar{S}_{0,0}$
1	0	$\bar{C}_{1,0}$	$\bar{S}_{1,0}$
1	1	$\bar{C}_{1,1}$	$\bar{S}_{1,1}$
2	0	$\bar{C}_{2,0}$	$\bar{S}_{2,0}$
2	1	$\bar{C}_{2,1}$	$\bar{S}_{2,1}$
2	2	$\bar{C}_{2,2}$	$\bar{S}_{2,2}$
3	0	$\bar{C}_{3,0}$	$\bar{S}_{3,0}$
3	1	$\bar{C}_{3,1}$	$\bar{S}_{3,1}$
3	2	$\bar{C}_{3,2}$	$\bar{S}_{3,2}$
3	3	$\bar{C}_{3,3}$	$\bar{S}_{3,3}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
N	$N-1$	$\bar{C}_{N,N-1}$	$\bar{S}_{N,N-1}$
N	N	$\bar{C}_{N,N}$	$\bar{S}_{N,N}$

We refer to this table as the **normalized coefficient table**. In many cases, one or more of the first three rows may be missing, and you may need to add this data manually for easy use later. This is because for any gravity model, the normalized coefficients for when both $n \leq 1$ and $m \leq 1$ are

---

[4]   Note that mean-tide models are not defined. This is partly because the gravitational effects of external bodies are typically considered separately; see Section **??**.

n	m	$\bar{C}_{nm}$	$\bar{S}_{nm}$
0	0	1	0
1	0	0	0
1	1	0	0

For more information, see Section 11.1.2 which discusses "special" gravitational coefficients.

Recall From Section 11.1 that for a given value of n, we always have that $0 \leq m \leq n$; this can also be seen in the table above. Therefore, for a given value of n, there are $n + 1$ rows of data. Let L be the number of rows of the normalized coefficient table. Then for a gravity model of maximum degree N, the number of rows are

$$L = \sum_{n=0}^{N}(n + 1)$$

This summation evaluates to[5]

$$\boxed{L = \frac{(N + 1)(N + 2)}{2}} \tag{11.13}$$

We can store the $\bar{C}_{n,m}$ and $\bar{S}_{n,m}$ columns of the coefficient table in the **normalized coefficient vectors** $\bar{\mathbf{C}} \in \mathbb{R}^L$ and $\bar{\mathbf{S}} \in \mathbb{R}^L$.

$$\bar{\mathbf{C}} = \begin{bmatrix} \bar{C}_{0,0} \\ \bar{C}_{1,0} \\ \bar{C}_{1,1} \\ \bar{C}_{2,0} \\ \bar{C}_{2,1} \\ \bar{C}_{2,2} \\ \bar{C}_{3,0} \\ \bar{C}_{3,1} \\ \bar{C}_{3,2} \\ \bar{C}_{3,3} \\ \vdots \\ \bar{C}_{N,N-1} \\ \bar{C}_{N,N} \end{bmatrix} \in \mathbb{R}^L, \qquad \bar{\mathbf{S}} = \begin{bmatrix} \bar{S}_{0,0} \\ \bar{S}_{1,0} \\ \bar{S}_{1,1} \\ \bar{S}_{2,0} \\ \bar{S}_{2,1} \\ \bar{S}_{2,2} \\ \bar{S}_{3,0} \\ \bar{S}_{3,1} \\ \bar{S}_{3,2} \\ \bar{S}_{3,3} \\ \vdots \\ \bar{S}_{N,N-1} \\ \bar{S}_{N,N} \end{bmatrix} \in \mathbb{R}^L \tag{11.14}$$

Alternatively, we could have stored them in $(N + 1) \times (N + 1)$ matrices, but that would result in a more inefficient storage method since about half of each matrix would be just zeros.

We say that the (n, m) coefficient is stored in the ℓth element of a coefficient vector. We refer to this index, ℓ, as the **gravity coefficient index**. To obtain the gravity coefficient index, recall that for a given value of n, there are $n + 1$ coefficients. Additionally, given values of n and m, the corresponding coefficient is the $(m + 1)$th element of the group of coefficients corresponding to the given value of n. Thus, to get ℓ, we first find the number of rows before the rows corresponding to the given n, and then add m to that number of rows.

$$\ell = m + \sum_{n=0}^{n-1}(n + 1)$$

The summation evaluates to[6]

$$\sum_{n=0}^{n-1}(n + 1) = \frac{n(n + 1)}{2}$$

---

[5] See https://www.wolframalpha.com/input?i=Sum+of+n%2B1+from+n%3D0+to+N.

[6] See https://www.wolframalpha.com/input?i=Sum+of+n%2B1+from+n%3D0+to+n-1.

so we have

$$\ell = \frac{n(n+1)}{2} + m$$

However, this equation is valid for 0-based indexing (to see this, we can check that for $n = m = 0$ we get $\ell = 0$). For 1-based indexing, we have to add 1 to this result, so we arrive at

$$\ell = \begin{cases} \dfrac{n(n+1)}{2} + m, & \text{0-based indexing} \\[3mm] \dfrac{n(n+1)}{2} + m + 1, & \text{1-based indexing} \end{cases} \tag{11.15}$$

Algorithm 121: `gravity_coeff_index`
Gravity coefficient index for a given degree and order.

Inputs:
- $n \in \mathbb{Z}$ - degree
- $m \in \mathbb{Z}$ - order

Procedure:

$$\ell = \begin{cases} \dfrac{n(n+1)}{2} + m, & \text{0-based indexing} \\[3mm] \dfrac{n(n+1)}{2} + m + 1, & \text{1-based indexing} \end{cases}$$

Outputs:
- $\ell \in \mathbb{Z}$ - gravity coefficient index

Test Cases:
- See Appendix **??**.

11.3   Obtaining the Normalized Gravitational Coefficients

The C_{nm} and S_{nm} coefficients are typically given[7] in a normalized form as $\bar{C}_{nm}$ and $\bar{S}_{nm}$, respectively. These normalizations are defined in Section **??**. The data files storing the coefficients may have different extensions (such as `.GEO` and `.ICGEM` for the GGM05S and GGM05C models), but these files can be opened as text (`.txt`) files. The files themselves have the data in a tabular form. The first four columns of the table are typically given as

---

[7]   [36] has a list of sources for gravitational coefficient data. The data for the various GGM05 models can be found at `http://download.csr.u texas.edu/pub/grace/GGM05/`.

n	m	$\bar{C}_{nm}$	$\bar{S}_{nm}$
0	0	$\bar{C}_{0,0}$	$\bar{S}_{0,0}$
1	0	$\bar{C}_{1,0}$	$\bar{S}_{1,0}$
1	1	$\bar{C}_{1,1}$	$\bar{S}_{1,1}$
2	0	$\bar{C}_{2,0}$	$\bar{S}_{2,0}$
2	1	$\bar{C}_{2,1}$	$\bar{S}_{2,1}$
2	2	$\bar{C}_{2,2}$	$\bar{S}_{2,2}$
3	0	$\bar{C}_{3,0}$	$\bar{S}_{3,0}$
3	1	$\bar{C}_{3,1}$	$\bar{S}_{3,1}$
3	2	$\bar{C}_{3,2}$	$\bar{S}_{3,2}$
3	3	$\bar{C}_{3,3}$	$\bar{S}_{3,3}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
N	$N-1$	$\bar{C}_{N,N-1}$	$\bar{S}_{N,N-1}$
N	N	$\bar{C}_{N,N}$	$\bar{S}_{N,N}$

In some cases, one or more of the first three rows may be missing, and you may need to add this data manually for easy use later. This is because for any gravity model, the normalized coefficients for when both $n \leq 1$ and $m \leq 1$ are

n	m	$\bar{C}_{nm}$	$\bar{S}_{nm}$
0	0	1	0
1	0	0	0
1	1	0	0

Since we have that $0 \leq n \leq N$ and $0 \leq m \leq N$, we can store the normalized coefficients in the matrices

$$\bar{\mathbf{C}} \in \mathbb{R}^{(N+1)\times(N+1)}$$

$$\bar{\mathbf{S}} \in \mathbb{R}^{(N+1)\times(N+1)}$$

Let `C_norm` and `S_norm` be arrays that store $\bar{\mathbf{C}}$ and $\bar{\mathbf{S}}$ in a computer, respectively. For programming languages with 0-based indexing, we can simply use the syntax

```
C_norm(n,m)
S_norm(n,m)
```

to access coefficients of degree n and order m. However, for programming languages with 1-based indexing (such as MATLAB), we must use

```
C_norm(n+1,m+1)
S_norm(n+1,m+1)
```

Algorithm 122: `coefftab2mat`
Coefficient table to coefficient matrices from a gravity data file.

Inputs:
- `path`   - file path to gravity data file
- $\mathbf{T} \in \mathbb{R}^{L\times 4}$ - coefficient table

Procedure:
1. Round the first two columns to ensure they are integers.
2. Preallocate the matrices $\bar{\mathbf{C}} \in \mathbb{R}^{(N+1)\times(N+1)}$ and $\bar{\mathbf{S}} \in \mathbb{R}^{(N+1)\times(N+1)}$ to store

the normalized gravitational coefficients.

$$\bar{\mathbf{C}} = \mathbf{0}_{(N+1)\times(N+1)}$$
$$\bar{\mathbf{S}} = \mathbf{0}_{(N+1)\times(N+1)}$$

3. Determine L given that $\mathbf{T} \in \mathbb{R}^{L\times 4}$.
4. Populate the coefficient matrices.

> **for** $i = 1$ **to** L
>
>> (a) Degree and order. *This step assumes 1-based indexing.*
>>
>> $$n = T_{i,1}$$
>> $$m = T_{i,1}$$
>>
>> (b) Terminate the loop if the maximum degree/order has been reached
>>
>>> **if** $(n > N)$
>>> | **break**
>>> **end**
>>
>> (c) Store the normalized gravitational coefficients. *This step assumes 1-based indexing.*
>>
>> $$\bar{\mathbf{C}}_{n+1,m+1} = T_{i,3}$$
>> $$\bar{\mathbf{S}}_{n+1,m+1} = T_{i,3}$$
>
> **end**

Outputs:
- $\mathbf{C} \in \mathbb{R}^{(N+1)\times(N+1)}$ - gravity coefficients
- $\mathbf{S} \in \mathbb{R}^{(N+1)\times(N+1)}$ - gravity coefficients

In the *Astrodynamic Toolbox*, the raw data file `GGM05S.ICGEM` can be found in the `toolbox/data/rawdata` folder. Since MATLAB can't read `.ICGEM` files, it we first saved the file as the text file `GGM05S.txt`. This file can also be found in the `toolbox/data/rawdata` folder. To load the coefficients for the GGM05s gravity model, use

```
GGM05S = load_GGM05S
```

`GGM05S` is a structure with two fields: `C_norm` ($(N+1) \times (N+1)$ matrix storing the $\bar{C}_{nm}$ coefficients) and `S_norm` ($(N+1) \times (N+1)$ matrix storing the $\bar{S}_{nm}$ coefficients). These can then be passed to `gravity_coeffs` (see Section **??**) to obtain the un-normalized gravity coefficients.

The `load_GGM05S` function loads the data from `toolbox/data/datafiles/GGM05S.mat`. This data file was created by running the script `toolbox/data/create/create_GGM05S.m`.

TODO: redo section above

The coefficients and deriving parameters of gravity models for many bodies in our solar system can be found at https://pds-geosciences.wustl.edu/dataserv/gravity_models.htm and https://ai-solutions.com/_help_Files/gravity_model_files.htm.

11.3.1   Earth Gravity Model: EGM2008

The latest Earth gravity model to be officially released[8] by the National Geospatial-Intelligence Agency (NGA) is the **Earth Gravitational Model 2008 (EGM2008)**. This model was developed from the combination of data from NASA's Gravity Recovery and Climate Experiment (GRACE[9]) mission with additional surface gravimetry and satellite altimetry data [67], [68, p. 79]. The EGM2008 model forms the core of the geopotential model that the 2010 IERS conventions recommend (additional details regarding the IERS recommendations can be found at the end of this section) [68, p. 79].

normalized	yes
tide model	tide-free [26, 36]
coordinate frame	ITRS[10]
standard gravitational parameter, $\mu_\oplus$ [m^3/s^2]	3986004.415×10^8 [26]
equatorial radius, $R_\oplus$ [m]	6378136.3 [26]
maximum degree	2190
maximum order	2159
data file	`EGM2008_to2190_TideFree` downloaded from[11] https://earth-info.nga.mil/php/download.php?file=egm-08spherical
data file description	`README_WGS84_2.pdf` (included with the data file download linked above)

In the data file linked above, the coefficient table has six columns. The first four columns of the coefficient table are formatted as follows:

n	m	$\bar{C}_{nm}$	$\bar{S}_{nm}$
2	0	$\bar{C}_{2,0}$	$\bar{S}_{2,0}$
2	1	$\bar{C}_{2,1}$	$\bar{S}_{2,1}$
⋮	⋮	⋮	⋮

Thus, we need to manually add the first three rows, as defined by Table 11.1 in Section 11.1.2:

n	m	$\bar{C}_{nm}$	$\bar{S}_{nm}$
0	0	1	0
1	0	0	0
1	1	0	0

---

[8] EGM2020 has been developed but has not yet been released.

[9] There are a set of gravity models, the GRACE gravity models, that were developed with data from the GRACE mission, and in some cases with additional data from the GOCE mission. https://www2.csr.utexas.edu/grace/gravity/ provides an overview of the GRACE models, while the data files are available at https://filedrop.csr.utexas.edu/pub/grace/GGM05/.

[11] This download can also be accessed by selecting "EGM2008" on the "Data/Apps" drop-down menu on https://earth-info.nga.mil/index.php?dir=wgs84&action=wgs84.

Incorporating Tidal Effects Using the IERS Conventional Model

The EGM2008 model is a tide-free model, implying that it does *not* include the permanent deformation of the Earth due to tidal effects. The IERS conventional model for the geopotential is at its core just the EGM2008, but includes some modifications and then applies tidal effects. First, the IERS conventional model modifies the $\bar{C}_{2,0}$, $\bar{C}_{3,0}$, $\bar{C}_{4,0}$, $\bar{C}_{2,1}$, and $\bar{S}_{2,1}$ normalized gravitational coefficients in Section 6.1 [68, pp. 79–81]. The procedure for incorporating tidal effects is then outlined through the remainder of Chapter 6 of the 2010 IERS conventions [68, pp. 81–96].

11.3.2   Lunar Gravity Model: GRGM1200A

The latest lunar gravity model produced by the United States is the degree and order 1200 GRGM1200A model developed from the results of the Gravity Recovery and Interior Laboratory (GRAIL) mission [52].

normalized	yes
tide model	tide-free [36]
coordinate frame	lunar body-fixed coordinate system in the principal axes frame as defined by the DE430 ephemeris[12]
standard gravitational parameter, $\mu_{\mathbb{C}}$ [m^3/s^2]	$4.9028001224453001 \times 10^3$
equatorial radius, $R_{\mathbb{C}}$ [m]	1738
maximum degree/order	1200
data file	https://pds-geosciences.wustl.edu/grail/grail-l-lgrs-5-rdr-v1/grail_1001/shadr/gggrx_1200a_sha.tab
data file description	https://pds-geosciences.wustl.edu/grail/grail-l-lgrs-5-rdr-v1/grail_1001/shadr/gggrx_1200a_sha.lbl

In the data file linked above, the first row is an 8 column wide header. The coefficient table starts at the second row. The first four columns of the coefficient table are formatted as follows:

n	m	$\bar{C}_{nm}$	$\bar{S}_{nm}$
1	0	$\bar{C}_{1,0}$	$\bar{S}_{1,0}$
1	1	$\bar{C}_{1,1}$	$\bar{S}_{1,1}$
2	0	$\bar{C}_{2,0}$	$\bar{S}_{2,0}$
2	1	$\bar{C}_{2,1}$	$\bar{S}_{2,1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Thus, we need to manually add the first row:

n	m	$\bar{C}_{nm}$	$\bar{S}_{nm}$
0	0	1	0

> Other lunar gravity models produced from data from the GRAIL mission can be found at https://pds-geosciences.wustl.edu/grail/grail-l-lgrs-5-rdr-v

---

[12] This coordinate frame is indicated in the data file description linked in this same table.

```
1/grail_1001/shadr/.
```

11.4 Obtaining Unnormalized Gravitational Coefficients

Recall Eqs. (11.7) and (11.8) from Section 11.1.3.

$$\bar{C}_{nm} = \frac{C_{nm}}{N_{nm}} \tag{11.7}$$

$$\bar{S}_{nm} = \frac{S_{nm}}{N_{nm}} \tag{11.8}$$

Therefore, if we are given the normalized coefficients $\bar{C}_{nm}$ and $\bar{S}_{nm}$, we can obtained their un-normalized counterparts C_{nm} and S_{nm} by rearranging these equations.

$$\boxed{C_{nm} = \frac{\bar{C}_{nm}}{N_{nm}}, \qquad S_{nm} = \frac{\bar{S}_{nm}}{N_{nm}}} \tag{11.16}$$

To obtain C_{nm} and S_{nm} using Eq. (11.16), we need to evaluate the normalization factor, N_{nm}, using Eq. (11.9) from Section 11.1.3.

$$N_{nm} = \sqrt{\frac{(n-m)!(2n+1)(2-\delta_{0,m})}{(n+m)!}} \tag{11.9}$$

However, evaluating Eq. (11.9) can lead to computational issues when $n + m > 170$ since 171! results in overflow when using IEEE double precision numbers. To avoid having to directly evaluate $(n+m)!$, we can compute N_{nm} in a recursive manner. There are three separate recursive equations that will be of use:

$$\boxed{N_{i,j-1} = N_{i,j}\sqrt{\frac{2-\delta_{0,j}}{(i+j)(i-j+1)}}} \tag{11.17}$$

$$\boxed{N_{i+1,j} = N_{i,j}\sqrt{\frac{(i+j+1)(2i+1)}{(i-j+1)(2i+3)}}} \tag{11.18}$$

$$\boxed{N_{i,j+1} = N_{i,j}\sqrt{\frac{(i+j+1)(i-j)}{2-\delta_{0,j}}}} \tag{11.19}$$

Let $\bar{C}$ and $\bar{S}$ be the $(N+1) \times (N+1)$ matrices storing the normalized coefficients up to maximum degree/order N, as defined in the previous section (Section **??**). Note that since $m \leq n$, these matrices will be lower triangular matrices. Using Algorithm 123, we can obtain the matrices C and S storing the *un*-normalized coefficients [60, p. 58][103, p. 546].

> **Algorithm 123: `gravity_coeffs`**
> Un-normalized gravity coefficients from the normalized gravity coefficients.
>
> ---
>
> **Inputs:**
> - $\bar{C} \in \mathbb{R}^{(N+1) \times (N+1)}$ - normalized gravity coefficients
> - $\bar{S} \in \mathbb{R}^{(N+1) \times (N+1)}$ - normalized gravity coefficients

Procedure:
1. Determine N given that $\bar{\mathbf{C}} \in \mathbb{R}^{(N+1)\times(N+1)}$.
2. Denormalize the gravity coefficients.

    for $n = 0$ **to** N
        for $m = 0$ **to** n
            (a) Scaling coefficient.

$$\Pi_{nm} = \sqrt{\frac{(n+m)!}{(2-\delta_{0,m})(2n+1)(n-m)!}}$$

            (b) Denormalize coefficients.

$$C_{nm} = \frac{\bar{C}_{nm}}{\Pi_{nm}}$$

$$S_{nm} = \frac{\bar{S}_{nm}}{\Pi_{m,m}}$$

        end
    end

Outputs:
- $\mathbf{C} \in \mathbb{R}^{(N+1)\times(N+1)}$ - gravity coefficients
- $\mathbf{S} \in \mathbb{R}^{(N+1)\times(N+1)}$ - gravity coefficients

11.5   Evaluating the Legendre Polynomials via Recursion

If we define

$$V_{nm} = \left(\frac{R_\oplus}{r}\right)^{n+1} P_{nm}[\sin(\phi_{\text{gc}})] \cos(m\lambda)$$

$$W_{nm} = \left(\frac{R_\oplus}{r}\right)^{n+1} P_{nm}[\sin(\phi_{\text{gc}})] \sin(m\lambda)$$
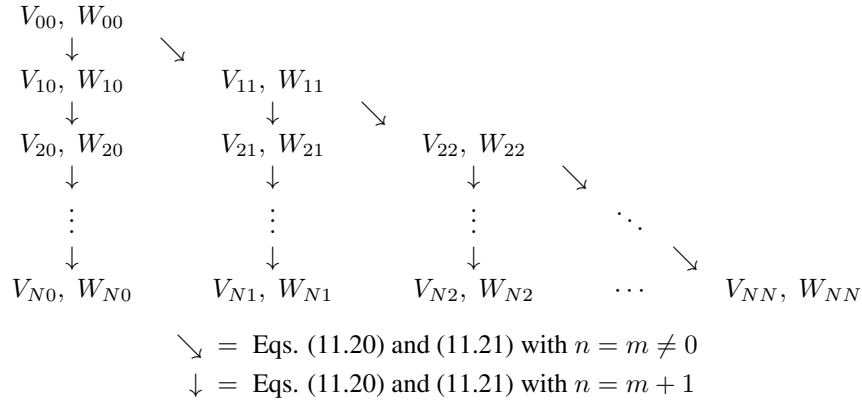
then the gravitational potential can be written as

$$U(r, \phi_{gc}, \lambda) = \frac{\mu_\oplus}{R_\oplus} \sum_{n=0}^{N} \sum_{m=0}^{n} (C_{nm}V_{nm} + S_{nm}W_{nm})$$

Note that we will not be using the above equation; instead, we will follow a procedure describes in the following section (Section **??**) to compute the gravitational acceleration directly. The coefficients V_{nm} and W_{nm} satisfy the recurrence relations

$$V_{nm} = \begin{cases} \dfrac{R_\oplus}{r}, & m = n = 0 \\[2ex] \dfrac{(2m-1)R_\oplus}{r^2}\left(r_X V_{m-1,m-1} - r_Y W_{m-1,m-1}\right), & m = n \neq 0 \\[2ex] \dfrac{r_Z R_\oplus}{r^2}\left(\dfrac{2n-1}{n-m}\right)V_{n-1,m} - \left(\dfrac{R_\oplus}{r}\right)^2\left(\dfrac{n+m-1}{n-m}\right)V_{n-2,m}, & n = m+1 \end{cases} \tag{11.20}$$

$$
W_{nm} = \begin{cases} 0, & m = n = 0 \\[2mm] \dfrac{(2m-1)R_\oplus}{r^2}\left(r_X W_{m-1,m-1} + r_Y V_{m-1,m-1}\right), & m = n \neq 0 \\[2mm] \dfrac{r_Z R_\oplus}{r^2}\left(\dfrac{2n-1}{n-m}\right)W_{n-1,m} - \left(\dfrac{R_\oplus}{r}\right)^2\left(\dfrac{n+m-1}{n-m}\right)W_{n-2,m}, & n = m+1 \end{cases} \tag{11.21}
$$

As evident through Eqs. (**??**)–(**??**) in the next section (Section **??**), we will need V_{nm} and W_{nm} up to and including degree/order $N+1$ to evaluate the gravitational acceleration using gravitational coefficients up to degree/order N. However, to compute partial derivatives of the gravitational acceleration, we will require V_{nm} and W_{nm} up to and including degree/order $N+2$ (see Section **??**). The resulting recursion (only up to degree/order N) can be depicted visually as [60, pp. 66–68]

$$
\begin{array}{ccccccc}
V_{00},\ W_{00} & & & & & & \\
\downarrow & \searrow & & & & & \\
V_{10},\ W_{10} & & V_{11},\ W_{11} & & & & \\
\downarrow & & \downarrow & \searrow & & & \\
V_{20},\ W_{20} & & V_{21},\ W_{21} & & V_{22},\ W_{22} & & \\
\vdots & & \vdots & & \vdots & \ddots & \\
\downarrow & & \downarrow & & \downarrow & & \searrow \\
V_{N0},\ W_{N0} & & V_{N1},\ W_{N1} & & V_{N2},\ W_{N2} & \cdots & V_{NN},\ W_{NN}
\end{array}
$$

$$\searrow \;=\; \text{Eqs. (11.20) and (11.21) with } n = m \neq 0$$
$$\downarrow \;=\; \text{Eqs. (11.20) and (11.21) with } n = m+1$$

Note that in some cases, the recursion tries to access values that aren't defined. For example, the calculation of V_{10} will try to use $V_{-1,0}$, which clearly does not exist. Similarly, for a given m, where $m > 0$, the calculation of the first sectorial term ($n = m+1$) will require $V_{n-2,m} = V_{m-1,m}$ and $W_{n-2,m} = W_{m-1,m}$, which similarly don't exist (as evident by the diagram above). To resolve this, we use 0 for all of the aforementioned values. For the case of calculating V_{10} and W_{10}, we need to do this explicitly. However, in its implementation, we will initialize the matrices as $\mathbf{V} = \mathbf{0}_{(N+2)\times(N+2)}$ and $\mathbf{W} = \mathbf{0}_{(N+2)\times(N+2)}$, so for the sectorial cases the $V_{m-1,m}$ and $W_{m-1,m}$ terms will already be defined as 0, and we don't have to explicitly account for these cases. Thus, the recursion can be performed using the following basic procedure:

1. Initialize the recursion by explicitly defining V_{00} and W_{00}.
2. Calculate the second zonal ($n = 1$, $m = 0$) terms using a modified form the $n = m+1$ case of Eqs. (11.20) and (11.21) with $m = 0$, $V_{-1,0} = 0$, and $W_{-1,0} = 0$

$$
\begin{aligned}
V_{10} &= \frac{r_Z R_\oplus}{r^2}\left(\frac{2n-1}{n}\right)V_{00} = \frac{r_Z R_\oplus}{r^2}V_{00} = \left(\frac{r_Z R_\oplus}{r^2}\right)\left(\frac{R_\oplus}{r}\right) \\
&= \frac{r_z R_\oplus^2}{r^3} \\
W_{10} &= \frac{r_Z R_\oplus}{r^2}W_{00} = \frac{r_Z R_\oplus}{r^2}(0) \\
&= 0
\end{aligned}
$$

3. Calculate the rest of the zonal terms (corresponding to $n \geq 2$ and $m = 0$) by looping through n from 2 to $N+1$ using the $n = m+1$ case of Eqs. (11.20) and (11.21). Note that substituting $m = 0$ into these cases results in

$$
\begin{aligned}
V_{n0} &= \frac{r_Z R_\oplus}{r^2}\left(\frac{2n-1}{n}\right)V_{n-1,0} - \left(\frac{R_\oplus}{r}\right)^2\left(\frac{n-1}{n}\right)V_{n-2,0} \\
W_{n0} &= \frac{r_Z R_\oplus}{r^2}\left(\frac{2n-1}{n}\right)W_{n-1,0} - \left(\frac{R_\oplus}{r}\right)^2\left(\frac{n-1}{n}\right)W_{n-2,0}
\end{aligned}
$$

Since $W_{00} = W_{10} = 0$, it follows that

$$W_{n0} = 0 \quad \forall \, n$$

4. Loop through m from 1 to $N + 1$:
   (a) Calculate the sectorial ($n = m$) terms using the $n = m \neq 0$ case of Eqs. (11.20) and (11.21).
   (b) Loop through n from $m + 1$ to $N + 1$, calculating the sectorial terms using the $n = m + 1$ case of Eqs. (11.20) and (11.21).

Algorithm 124 below formalizes this recursive procedure to aid its computational implementation. Note that some additional simplifications are made from the above procedure. Notably, since $\mathbf{W}$ is initialized as the zero matrix, we do not have to explicitly define W_{00} and W_{10}. Also note that since MATLAB uses 1-based indexing, it is more convenient to return functions V and W that will provide $V_{nm} = $ V(n,m) and $W_{nm} = $ W(n,m). Finally, some auxiliary parameters have been introduced to (a) simplify the equations and (b) avoid recalculating quantities unnecessarily.

Algorithm 124: `legendre_recursion`

Recursive evaluation of coefficients related to the Legendre polynomials required for evaluating the gravitational acceleration.

Inputs:
- $[\mathbf{r}]_{\text{ecef}} \in \mathbb{R}^3$ - position resolved in the ECEF frame [m]
- $R_{\oplus} \in \mathbb{R}$ - Earth mean equatorial radius [m]
- $N \in \mathbb{R}$ - maximum degree/order

Procedure:
1. Magnitude of ECEF position squared [m²].

$$r^2 = [\mathbf{r}]_{\text{ecef}} \cdot [\mathbf{r}]_{\text{ecef}}$$

2. Distance from center of the Earth [m].

$$r = \sqrt{r^2}$$

3. Auxiliary parameter.

$$a = \frac{R_{\oplus}^2}{r^2}$$

4. Extract r_X, r_Y, and r_Z from $[\mathbf{r}]_{\text{ecef}} = (r_X, r_Y, r_Z)^T$.
5. "Normalized" ECEF position components.

$$x = \frac{R_{\oplus} r_X}{r^2}$$
$$y = \frac{R_{\oplus} r_Y}{r^2}$$
$$z = \frac{R_{\oplus} r_Z}{r^2}$$

6. Initialize $\mathbf{V}$ and $\mathbf{W}$ as zero matrices.

$$\mathbf{V} = \mathbf{0}_{(N+3) \times (N+3)}$$
$$\mathbf{W} = \mathbf{0}_{(N+3) \times (N+3)}$$

7. First zonal term ($n = m = 0$). Note that $W_{00} = 0$ is already handled by the previous step.

$$V_{00} = \frac{R_{\oplus}}{r}$$

8. Second zonal terms ($n = 1$, $m = 0$). Note that $W_{10} = 0$ is already handled by step 3.

$$V_{10} = zV_{00}$$

9. Remaining zonal terms ($2 \leq n \leq N + 1$, $m = 0$). Note that $W_{n0} = 0 \ \forall \ n$, and since we initialized $\mathbf{W}$ as a zero matrix, we do not have to perform any calculations here for W_{n0}.

    for $n = 2$ **to** $N + 2$
$$V_{n0} = \frac{z(2n - 1)V_{n-1,0} - a(n - 1)V_{n-2,0}}{n}$$
    end

10. Find V_{nm} and W_{nm} for all $n, m \in [1, N]$ (tesseral and sectorial terms).

    for $m = 1$ **to** $N + 2$
        (a) Sectorial terms ($n = m$).

$$V_{mm} = (2m - 1)\left(xV_{m-1,m-1} - yW_{m-1,m-1}\right)$$
$$W_{mm} = (2m - 1)\left(xW_{m-1,m-1} + yV_{m-1,m-1}\right)$$

        (b) Tesseral terms ($n > m$).

        for $n = m + 1$ **to** $N + 2$
$$V_{nm} = \frac{z(2n - 1)V_{n-1,m} - a(n + m - 1)V_{n-2,m}}{n - m}$$
$$W_{nm} = \frac{z(2n - 1)W_{n-1,m} - a(n + m - 1)W_{n-2,m}}{n - m}$$
        end

    end

Outputs:
- $\mathbf{V} \in \mathbb{R}^{(N+2) \times (N+2)}$ - Legendre coefficients
- $\mathbf{W} \in \mathbb{R}^{(N+2) \times (N+2)}$ - Legendre coefficients

11.6   Computation of the Gravitational Acceleration

Since a gravitational field is conservative, the acceleration due to the Earth's gravitational potential may be written as

$$\mathbf{a} = \nabla U$$

However, we want to express the acceleration in the ECEF frame (i.e. in Cartesian coordinates). [103, p. 550] details how to find this gradient via the chain rule, and lists all relevant partial derivatives and Jacobians.

Let

$$[\mathbf{g}]_{\text{ecef}} = \begin{bmatrix} g_X \\ g_Y \\ g_Z \end{bmatrix} \tag{11.22}$$

be the *inertial* gravitational acceleration resolved in the ECEF frame. Then

$$
\begin{aligned}
g_X &= \sum_{n=0}^{N}\sum_{m=0}^{n}(g_X)_{nm} \\
g_Y &= \sum_{n=0}^{N}\sum_{m=0}^{n}(g_Y)_{nm} \\
g_Z &= \sum_{n=0}^{N}\sum_{m=0}^{n}(g_Z)_{nm}
\end{aligned}
\tag{11.23}
$$

where $(g_{(\cdot)})_{nm}$ represents the **partial gravitational acceleration** due to the coefficients with degree n and order m. The partial accelerations can be computed as [60, p. 68]

$$
(g_X)_{nm} =
\begin{cases}
\dfrac{\mu_\oplus}{R_\oplus^2}\left(-C_{n,0}V_{n+1,1}\right), & m=0 \\[2ex]
\dfrac{\mu_\oplus}{2R_\oplus^2}\bigg\{ (-C_{nm}V_{n+1,m+1} - S_{nm}W_{n+1,m+1}) & m>0 \\
\qquad + \left[\dfrac{(n-m+2)!}{(n-m)!}\right](C_{nm}V_{n+1,m-1} + S_{nm}W_{n+1,m-1})\bigg\},
\end{cases}
$$

$$
(g_Y)_{nm} =
\begin{cases}
\dfrac{\mu_\oplus}{R_\oplus^2}\left(-C_{n,0}W_{n+1,1}\right), & m=0 \\[2ex]
\dfrac{\mu_\oplus}{2R_\oplus^2}\bigg\{ (-C_{nm}W_{n+1,m+1} + S_{nm}V_{n+1,m+1}) & m>0 \\
\qquad + \left[\dfrac{(n-m+2)!}{(n-m)!}\right](-C_{nm}W_{n+1,m-1} + S_{nm}V_{n+1,m-1})\bigg\},
\end{cases}
$$

$$
(g_Z)_{nm} = \frac{\mu_\oplus}{R_\oplus^2}\left[(n-m+1)(-C_{nm}V_{n+1,m} - S_{nm}W_{n+1,m})\right]
$$

However, we can simplify these equations by introducing the following auxiliary parameters:

$$
a = \frac{\mu_\oplus}{2R_\oplus^2}
\tag{11.24}
$$

$$
b_{nm} = \frac{(n-m+2)!}{(n-m)!}
$$

The second auxiliary parameter can be simplified[13] to

$$
b_{nm} = (m-n-2)(m-n-1)
\tag{11.25}
$$

The partial gravitational accelerations can then be written as

$$
(g_X)_{nm} =
\begin{cases}
-2aC_{n,0}V_{n+1,1}, & m=0 \\
-a(C_{nm}V_{n+1,m+1} + S_{nm}W_{n+1,m+1}) + ab_{nm}(C_{nm}V_{n+1,m-1} + S_{nm}W_{n+1,m-1}) & m>0
\end{cases}
\tag{11.26}
$$

$$
(g_Y)_{nm} =
\begin{cases}
-2aC_{n,0}W_{n+1,1}, & m=0 \\
a(S_{nm}V_{n+1,m+1} - C_{nm}W_{n+1,m+1}) + ab_{nm}(S_{nm}V_{n+1,m-1} - C_{nm}W_{n+1,m-1}) & m>0
\end{cases}
\tag{11.27}
$$

---

[13] See `mlx_factorial_simplification.mlx` in the `derivations/forcemodels` folder of the *Astrodynamics Toolbox*.

$$(g_Z)_{nm} = 2a(n - m + 1)(-C_{nm}V_{n+1,m} - S_{nm}W_{n+1,m})$$

Note that $S_{n0} = 0$. Therefore,

$$(g_Z)_{n0} = -2a(n + 1)C_{n0}V_{n+1,0}$$

and we can write

$$(g_Z)_{nm} = \begin{cases} -2a(n + 1)C_{n0}V_{n+1,0}, & m = 0 \\ -2a(n - m + 1)(C_{nm}V_{n+1,m} + S_{nm}W_{n+1,m}), & m > 0 \end{cases} \tag{11.28}$$

Algorithm 125 implements the calculation of the gravitational acceleration.

Algorithm 125: `gravity`

Gravitational acceleration using a spherical harmonic model up to order/degree N.

Inputs:
- $[\mathbf{r}]_{\text{ecef}} \in \mathbb{R}^3$ - position resolved in ECEF frame $[\text{m}]$
- $\mathbf{C} \in \mathbb{R}^{(N+1)\times(N+1)}$ - un-normalized gravitational coefficients
- $\mathbf{S} \in \mathbb{R}^{(N+1)\times(N+1)}$ - un-normalized gravitational coefficients
- $N \in \mathbb{R}$ - maximum degree/order of gravity model to use

Procedure:
1. Define $\mu_{\oplus}$ $[\text{m}^3/\text{s}^2]$, and $R_{\oplus}$ $[\text{m}]$ using the values from Appendix **??** (these values can be found in `GGM05S.txt` in the `toolbox/data/rawdata` folder).
2. Recursively compute the Legendre coefficients using Algorithm 124.

$$[\mathbf{V}, \mathbf{W}] = \texttt{legendre\_recursion}([\mathbf{r}]_{\text{ecef}}, R_{\oplus}, N)$$

3. Auxiliary parameter $[\text{m}/\text{s}^2]$.

$$a = \frac{\mu_{\oplus}}{2R_{\oplus}^2}$$

4. Initialize the accelerations along the three axes of the ECEF frame. $[\text{m}/\text{s}^2]$

$$g_X = 0$$
$$g_Y = 0$$
$$g_Z = 0$$

5. Calculate the accelerations.

    for $n = 0$ **to** N

(a) Zonal terms ($m = 0$).

$$g_X = g_X - 2aC_{n,0}V_{n+1,1}$$
$$g_Y = g_Y - 2aC_{n,0}W_{n+1,1}$$
$$g_Z = g_Z - 2a(n+1)C_{n0}V_{n+1,0}$$

(b) Sectorial ($m = n$) and tesseral ($n > m$) terms.

for $m = 1$ **to** n

$b = (m - n - 2)(m - n - 1)$

$g_X = g_X - a(C_{nm}V_{n+1,m+1} + S_{nm}W_{n+1,m+1})$
$\qquad + ab(C_{nm}V_{n+1,m-1} + S_{nm}W_{n+1,m-1})$

$g_Y = g_Y + a(S_{nm}V_{n+1,m+1} - C_{nm}W_{n+1,m+1})$
$\qquad + ab(S_{nm}V_{n+1,m-1} - C_{nm}W_{n+1,m-1})$

$g_Z = g_Z - 2a(n - m + 1)(C_{nm}V_{n+1,m}$
$\qquad + S_{nm}W_{n+1,m})$

end

end

6. Gravitational acceleration resolved in ECEF frame.

$$[\mathbf{g}]_{\text{ecef}} = \begin{bmatrix} g_X \\ g_Y \\ g_Z \end{bmatrix}$$

Outputs:

- $[\mathbf{g}]_{\text{ecef}} \in \mathbb{R}^{N \times N}$ - gravitational acceleration resolved in ECEF frame $[\text{m/s}^2]$

11.6.1   Simplified Model: Point Mass Gravity

TODO: use table 1.1. from iers 36 (2010 petit) TODO: rename to point mass gravity Recall the FODE from Eq. (**??**) from Section **??**:

$$\ddot{\mathbf{r}} = -\mu_\oplus \frac{\mathbf{r}}{r^3}$$

The FODE defines the equation of motion for the two-body problem, which assumes[14] a spherical central body. The solution of this two-body problem is a Keplerian orbit; hence, we refer to the right hand side of the equation above as the **two-body gravitational acceleration** or the **Keplerian gravitational acceleration**.

$$\boxed{\mathbf{g}_{\text{kep}} = -\mu_\oplus \frac{\mathbf{r}}{r^3}} \tag{11.29}$$

---

[14] See Section **??** for a full summary of the assumptions.

Algorithm 126: `two_body`
Two-body/Keplerian gravitational acceleration.

Inputs:
- $\mathbf{r} \in \mathbb{R}^3$ - position $[\mathrm{m}]$

Procedure:
1. Define $\mu_\oplus$ $[\mathrm{m}^3/\mathrm{s}^2]$ using the value from Appendix **??**.
2. Two-body/Keplerian gravitational acceleration $[\mathrm{m}/s^2]$.

$$\mathbf{g}_{\mathrm{kep}} = -\mu_\oplus \frac{\mathbf{r}}{r^3}$$

Outputs:
- $\mathbf{g}_{\mathrm{kep}} \in \mathbb{R}^3$ - Two-body/Keplerian gravitational acceleration $[\mathrm{m}/s^2]$

Note:
- $\mathbf{r}$ must be defined in a geocentric coordinate frame.

11.6.2   Simplified Model: J2 Gravity

In the Earth-centered Earth-fixed (ECEF) frame, the J_2 perturbation can be expressed as [23, p. 664], [91, p. 32]

$$[\mathbf{f}_{J_2}]_{\mathrm{ecef}} = \frac{3J_2\mu_\oplus R_\oplus^2}{2r^4} \begin{bmatrix} \dfrac{r_X}{r}\left(\dfrac{5r_Z^2}{r^2} - 1\right) \\[2mm] \dfrac{r_Y}{r}\left(\dfrac{5r_Z^2}{r^2} - 1\right) \\[2mm] \dfrac{r_Z}{r}\left(\dfrac{5r_Z^2}{r^2} - 3\right) \end{bmatrix} \tag{11.30}$$

where $[\mathbf{r}]_{\mathrm{ecef}} = (r_X, r_Y, r_Z)^T$ is the position of the satellite in the ECEF frame, and $r = \sqrt{r_X^2 + r_Y^2 + r_Z^2}$.

Algorithm 127: `J2_ecef`
Perturbing gravitational acceleration due to J_2 (in the ECEF frame).

Inputs:
- $[\mathbf{r}]_{\mathrm{ecef}} \in \mathbb{R}^3$ - position resolved in ECEF frame $[\mathrm{m}]$

Procedure:
1. Define J_2, $\mu_\oplus$ $[\mathrm{m}^3/\mathrm{s}^2]$, and $R_\oplus$ $[\mathrm{m}]$ using the values from Appendix **??**.
2. Extract r_X, r_Y, and r_Z from $[\mathbf{r}]_{\mathrm{ecef}}$.
3. Distance from center of the Earth $[\mathrm{m}]$.

$$\mathbf{r} = \sqrt{r_X^2 + r_Y^2 + r_Z^2}$$

4. Perturbing gravitational acceleration due to J_2 $[\text{m/s}^2]$.

$$[\mathbf{f}_{J_2}]_{\text{ecef}} = \frac{3J_2\mu_\oplus R_\oplus^2}{2r^4} \begin{bmatrix} \dfrac{r_X}{r}\left(\dfrac{5r_Z^2}{r^2}-1\right) \\[2ex] \dfrac{r_Y}{r}\left(\dfrac{5r_Z^2}{r^2}-1\right) \\[2ex] \dfrac{r_Z}{r}\left(\dfrac{5Z^2}{r^2}-3\right) \end{bmatrix}$$

Outputs:
   • $[\mathbf{f}_{J_2}]_{\text{ecef}} \in \mathbb{R}^3$ - perturbing gravitational acceleration due to J_2 resolved in the ECEF frame $[\text{m/s}^2]$

In the Hill/orbital (RTN) frame, the J_2 perturbation can be expressed as [23, p. 685]

$$[\mathbf{f}_{J_2}] = -\frac{3J_2\mu_\oplus R_\oplus^2}{2r^4} \begin{bmatrix} 1 - 3\sin^2(i)\sin^2(u) \\ \sin^2(i)\sin(u)\cos(u) \\ \sin(i)\cos(i)\sin(u) \end{bmatrix} \tag{11.31}$$

Algorithm 128: `J2_rtn`

Perturbing gravitational acceleration due to J_2 (in the RTN frame).

Inputs:
   • $r \in \mathbb{R}$ - distance between center of Earth and satellite $[\text{m}]$
   • $i \in \mathbb{R}$ - inclination $[\text{rad}]$
   • $u \in \mathbb{R}$ - argument of latitude $[\text{rad}]$

Procedure:
   1. Define J_2, $\mu_\oplus$ $[\text{m}^3/\text{s}^2]$, and $R_\oplus$ $[\text{m}]$ using the values from Appendix **??**.
   2. Perturbing gravitational acceleration due to J_2 $[\text{m/s}^2]$.

$$[\mathbf{f}_{J_2}] = -\frac{3J_2\mu_\oplus R_\oplus^2}{2r^4} \begin{bmatrix} 1 - 3\sin^2(i)\sin^2(u) \\ \sin^2(i)\sin(u)\cos(u) \\ \sin(i)\cos(i)\sin(u) \end{bmatrix}$$

Outputs:
   • $[\mathbf{f}_{J_2}] \in \mathbb{R}^3$ - perturbing gravitational acceleration due to J_2 resolved in the RTN frame $[\text{m/s}^2]$

11.7   Computation of the Gravity Gradient

Recall that in Eq. (**??**) in Section **??**, we defined the *inertial* gravitational acceleration resolved in the ECEF frame as

$$[\mathbf{g}]_{\text{ecef}} = \begin{bmatrix} g_X \\ g_Y \\ g_Z \end{bmatrix}$$

Ultimately, our goal is to find the partial derivatives of the gravitational acceleration with respect to position and inertial velocity, resolved in the ECI frame. However, since the gravitational models are implemented with respect to the ECEF frame, we first consider the partial derivatives with respect to the ECEF frame. To match the notation for the other sections, we define

$$\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\text{ecef}} = \left[\frac{\partial \mathbf{g}}{\partial \mathbf{r}}\right]_{\text{ecef}}$$

$$\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_g\right]_{\text{ecef}} = \left[\frac{\partial \mathbf{g}}{\partial \mathbf{v}}\right]_{\text{ecef}}$$

Immediately, we can note that since the gravitational acceleration is independent of velocity,

$$\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_g\right]_{\text{ecef}} = \mathbf{0}_{3 \times 3} \tag{11.32}$$

Our next goal is to find the partial derivative of the gravitational acceleration with respect to position, resolved in the ECEF frame. This partial derivative can be written in terms of its components as

$$\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\text{ecef}} = \begin{bmatrix} \dfrac{\partial g_X}{\partial X} & \dfrac{\partial g_X}{\partial Y} & \dfrac{\partial g_X}{\partial Z} \\[2mm] \dfrac{\partial g_Y}{\partial X} & \dfrac{\partial g_Y}{\partial Y} & \dfrac{\partial g_Y}{\partial Z} \\[2mm] \dfrac{\partial g_Z}{\partial X} & \dfrac{\partial g_Z}{\partial Y} & \dfrac{\partial g_Z}{\partial Z} \end{bmatrix} \tag{11.33}$$

A subset[15] of these partial derivatives are defined as

$$\begin{aligned} \frac{\partial g_X}{\partial X} &= \sum_{n=0}^{N} \sum_{m=0}^{n} \left(\frac{\partial g_X}{\partial X}\right)_{nm} \\ \frac{\partial g_X}{\partial Y} &= \sum_{n=0}^{N} \sum_{m=0}^{n} \left(\frac{\partial g_X}{\partial Y}\right)_{nm} \\ \frac{\partial g_X}{\partial Z} &= \sum_{n=0}^{N} \sum_{m=0}^{n} \left(\frac{\partial g_X}{\partial Z}\right)_{nm} \\ \frac{\partial g_Y}{\partial Z} &= \sum_{n=0}^{N} \sum_{m=0}^{n} \left(\frac{\partial g_Y}{\partial Z}\right)_{nm} \\ \frac{\partial g_Z}{\partial Z} &= \sum_{n=0}^{N} \sum_{m=0}^{n} \left(\frac{\partial g_Z}{\partial Z}\right)_{nm} \end{aligned} \tag{11.34}$$

---

[15] The remaining partial derivatives will be defined near the end of this section.

The degree n and order m terms (i.e. the $(\partial g_{(\cdot)}/\partial(\cdot))_{nm}$ terms) can be computed as [60, pp. 245–246]

$$
\left(\frac{\partial g_X}{\partial X}\right)_{nm} =
\begin{cases}
\dfrac{1}{2}\left(\dfrac{\mu_\oplus}{R_\oplus^3}\right)\left[C_{n0}V_{n+2,2} - \dfrac{(n+2)!}{n!}(C_{n0}V_{n+2,0})\right], & m = 0 \\[2ex]
\dfrac{1}{4}\left(\dfrac{\mu_\oplus}{R_\oplus^3}\right)\Big\{(C_{n1}V_{n+2,3} + S_{n1}W_{n+2,3}) & m = 1 \\[1ex]
\quad + \left[\dfrac{(n+1)!}{(n-1)!}\right](-3C_{n1}V_{n+2,1} - S_{n1}W_{n+2,1})\Big\}, & \\[2ex]
\dfrac{1}{4}\left(\dfrac{\mu_\oplus}{R_\oplus^3}\right)\Big\{(C_{nm}V_{n+2,m+2} + S_{nm}W_{n+2,m+2}) & m > 1 \\[1ex]
\quad + 2\left[\dfrac{(n-m+2)!}{(n-m)!}\right](-C_{nm}V_{n+2,m} - S_{nm}W_{n+2,m}) & \\[1ex]
\quad + \left[\dfrac{(n-m+4)!}{(n-m)!}\right](C_{nm}V_{n+2,m-2} + S_{nm}W_{n+2,m-2})\Big\}, &
\end{cases}
$$

$$
\left(\frac{\partial g_X}{\partial Y}\right)_{nm} =
\begin{cases}
\dfrac{1}{2}\left(\dfrac{\mu_\oplus}{R_\oplus^3}\right)(C_{n0}W_{n+2,2}), & m = 0 \\[2ex]
\dfrac{1}{4}\left(\dfrac{\mu_\oplus}{R_\oplus^3}\right)\Big\{(C_{n1}W_{n+2,3} - S_{n1}V_{n+2,3}) & m = 1 \\[1ex]
\quad + \left[\dfrac{(n+1)!}{(n-1)!}\right](-C_{n1}W_{n+2,1} - S_{n1}V_{n+2,1})\Big\}, & \\[2ex]
\dfrac{1}{4}\left(\dfrac{\mu_\oplus}{R_\oplus^3}\right)\Big\{(C_{nm}W_{n+2,m+2} - S_{nm}V_{n+2,m+2}) & m > 1 \\[1ex]
\quad + \left[\dfrac{(n-m+4)!}{(n-m)!}\right](-C_{nm}W_{n+2,m-2} + S_{nm}V_{n+2,m-2})\Big\}, &
\end{cases}
$$

$$
\left(\frac{\partial g_X}{\partial Z}\right)_{nm} =
\begin{cases}
\dfrac{\mu_\oplus}{R_\oplus^3}(n+1)(C_{n0}V_{n+2,1}), & m = 0 \\[2ex]
\dfrac{\mu_\oplus}{R_\oplus^3}\Big\{\dfrac{n-m+1}{2}(C_{nm}V_{n+2,m+1} + S_{nm}W_{n+2,m+1}) & m > 0 \\[1ex]
\quad + \left[\dfrac{(n-m+3)!}{2(n-m)!}\right](-C_{nm}V_{n+2,m-1} - S_{nm}W_{n+2,m-1})\Big\}, &
\end{cases}
\tag{11.35}
$$

$$
\left(\frac{\partial g_Y}{\partial Z}\right)_{nm} =
\begin{cases}
\dfrac{\mu_\oplus}{R_\oplus^3}(n+1)(C_{n0}W_{n+2,1}), & m = 0 \\[2ex]
\dfrac{\mu_\oplus}{R_\oplus^3}\Big\{\dfrac{n-m+1}{2}(C_{nm}W_{n+2,m+1} - S_{nm}V_{n+2,m+1}) & m > 0 \\[1ex]
\quad + \left[\dfrac{(n-m+3)!}{2(n-m)!}\right](C_{nm}W_{n+2,m-1} - S_{nm}V_{n+2,m-1})\Big\}, &
\end{cases}
\tag{11.36}
$$

$$
\left(\frac{\partial g_Z}{\partial Z}\right)_{nm} = \frac{\mu_\oplus}{R_\oplus^3}\left[\frac{(n-m+2)!}{(n-m)!}(C_{nm}V_{n+2,m} + S_{nm}W_{n+2,m})\right]
\tag{11.37}
$$

Note that the reason we precompute W_{nm} and V_{nm} up to terms with $N+2$ in Algorithm 124 (Section **??**) is because the equations above require terms up to degree $N+2$ [60, p. 246].

Moving on, we can simplify the equations themselves and their implementation by introducing the following

auxiliary parameters[16]:

$$a = \frac{\mu_\oplus}{R_\oplus^3}$$

$$b = n + 1$$

$$c = \frac{(n+2)!}{n!} = (n+1)(n+2)$$

$$d = \frac{(n+1)!}{(n-1)!} = n(n+1)$$

$$e = \frac{n-m+1}{2}$$

$$f = \frac{(n-m+2)!}{(n-m)!} = (m-n-2)(m-n-1)$$

$$g = \frac{(n-m+4)!}{(n-m)!} = (m-n-4)(m-n-3)(m-n-2)(m-n-1)$$

$$h = \frac{(n-m+3)!}{2(n-m)!} = -\frac{(m-n-3)(m-n-2)(m-n-1)}{2}$$

We can further simplify g and h by writing them in terms of f as

$$g = (m-n-4)(m-n-3)f$$

$$h = -\frac{(m-n-3)f}{2} = \frac{(3+n-m)f}{2}$$

Additionally, in the implementation of the equations in this section, all the parameters like C_{nm} and $V_{n+2,m}$ are manually accessing elements of arrays. Oftentimes, we would have access the same element multiple times, creating an unnecessary computational cost. Therefore, we define auxiliary variables for each of these terms as well.

$i = C_{nm}$	$j = S_{nm}$	$k = C_{n0}$	$l = V_{n+2,2}$	$o = V_{n+2,0}$
$p = C_{n1}$	$q = V_{n+2,3}$	$r = S_{n1}$	$s = W_{n+2,3}$	$t = V_{n+2,1}$
$u = W_{n+2,1}$	$v = V_{n+2,m+2}$	$w = W_{n+2,m+2}$	$x = V_{n+2,m}$	$y = W_{n+2,m}$
$z = V_{n+2,m-2}$	$\alpha = W_{n+2,m-2}$	$\beta = V_{n+2,m+1}$	$\gamma = W_{n+2,m+1}$	$\delta = V_{n+2,m-1}$
$\varepsilon = W_{n+2,m-1}$	$\zeta = W_{n+2,2}$			

> To save space and to make the equations look neater, these auxiliary parameters have not been subscripted with n or nm, even though they depend on the values of n and m. For more "formal" definitions of these parameters, they should be subscripted in both their definitions and when used in equations. However, since the main purpose of defining these auxiliary parameters is for computational implementation, we do not include the subscripts. Any time the value of n or m changes, we will simply update the value of these auxiliary parameters as well.

The derivatives can be written in terms of the auxiliary parameters as

$$\left(\frac{\partial g_X}{\partial X}\right)_{nm} = \begin{cases} \dfrac{a(kl - cko)}{2}, & m = 0 \\[3mm] \dfrac{a[pq + rs - d(3pt + ru)]}{4}, & m = 1 \\[3mm] \dfrac{a[iv + jw - 2f(ix + jy) + g(iz + j\alpha)]}{4}, & m > 1 \end{cases} \tag{11.38}$$

---

[16] The simplification of the factorials was done using https://www.wolframalpha.com/.

$$\left(\frac{\partial g_X}{\partial Y}\right)_{nm} = \begin{cases} \dfrac{ak\zeta}{2}, & m = 0 \\[2ex] \dfrac{a[ps - rq - d(pu + rt)]}{4}, & m = 1 \\[2ex] \dfrac{a[iw - jv + g(jz - i\alpha)]}{4}, & m > 1 \end{cases} \tag{11.39}$$

$$\left(\frac{\partial g_X}{\partial Z}\right)_{nm} = \begin{cases} abkt, & m = 0 \\ a[e(i\beta + j\gamma) - h(i\delta + j\varepsilon)], & m > 0 \end{cases}$$

$$\left(\frac{\partial g_Y}{\partial Z}\right)_{nm} = \begin{cases} abku, & m = 0 \\ a[e(i\gamma - j\beta) + h(i\varepsilon - j\delta)], & m > 0 \end{cases}$$

$$\left(\frac{\partial g_Z}{\partial Z}\right)_{nm} = af(ix + jy)$$

For implementation purposes, it is easier to also explicitly define $(\partial g_X/\partial Z)_{nm}$, $(\partial g_Y/\partial Z)_{nm}$, and $(\partial g_Z/\partial Z)_{nm}$ for the case when $m = 1$. For $(\partial g_X/\partial Z)_{n1}$, we have (from Eq. (11.35))

$$\left(\frac{\partial g_X}{\partial z}\right)_{n1} = \frac{\mu_\oplus}{R_\oplus^3} \left\{ \frac{n}{2}(C_{n1}V_{n+2,2} + S_{n1}W_{n+2,2}) + \left[\frac{(n+2)!}{2(n-1)!}\right](-C_{n1}V_{n+2,0} - S_{n1}W_{n+2,0}) \right\}$$

In terms of the existing auxiliary parameters, we can rewrite this equation as

$$\left(\frac{\partial g_X}{\partial z}\right)_{n1} = a\left\{ \frac{n(pl + r\zeta)}{2} - \left[\frac{(n-2)!}{(n-1)!}\right]\frac{po + rW_{n+2,0}}{2} \right\}$$

However, from Section **??**, we know that $W_{n0} = 0 \ \forall \ n$, so $W_{n+2,0} = 0$. Additionally, we define a new auxiliary parameter η as

$$\eta = \frac{(n+2)!}{(n-1)!} = n(n+1)(n+2)$$

We can then write $(\partial g_X/\partial Z)_{n1}$ as

$$\left(\frac{\partial g_X}{\partial z}\right)_{n1} = a\left[\frac{n(pl + r\zeta)}{2} - \frac{\eta po}{2}\right] = \frac{a[n(pl + r\zeta) - \eta po]}{2}$$

Using $(\partial g_X/\partial Z)_{n1}$, we redefine $(\partial g_X/\partial Z)_{nm}$ as

$$\left(\frac{\partial g_X}{\partial Z}\right)_{nm} = \begin{cases} abkt, & m = 0 \\[1ex] \dfrac{a[n(pl + r\zeta) - \eta po]}{2}, & m = 1 \\[1ex] a[e(i\beta + j\gamma) - h(i\delta + j\varepsilon)], & m > 1 \end{cases} \tag{11.40}$$

For $(\partial g_Y/\partial Z)_{n1}$, we have (from Eq. (11.36))

$$\left(\frac{\partial g_Y}{\partial z}\right)_{n1} = \frac{\mu_\oplus}{R_\oplus^3} \left\{ \frac{n}{2}(C_{n1}W_{n+2,2} - S_{n1}V_{n+2,2}) + \left[\frac{(n+2)!}{2(n-1)!}\right](C_{n1}W_{n+2,0} - S_{n1}V_{n+2,0}) \right\}$$

Rearranging slightly, and noting again that $W_{n+2,0} = 0$,

$$\left(\frac{\partial g_Y}{\partial z}\right)_{n1} = \frac{\mu_\oplus}{R_\oplus^3} \left\{ \frac{n(C_{n1}W_{n+2,2} - S_{n1}V_{n+2,2})}{2} - \left[\frac{(n+2)!}{(n-1)!}\right]\frac{S_{n1}V_{n+2,0}}{2} \right\}$$

In terms of existing auxiliary parameters, we can rewrite this equation as

$$\left(\frac{\partial g_Y}{\partial z}\right)_{n1} = a\left[\frac{n(p\zeta - rl)}{2} - \eta\left(\frac{ro}{2}\right)\right] = \frac{a[n(p\zeta - rl) - \eta ro]}{2}$$

Using $(\partial g_Y/\partial Z)_{n1}$, we redefine $(\partial g_Y/\partial Z)_{nm}$ as

$$\boxed{\left(\frac{\partial g_Y}{\partial Z}\right)_{nm} = \begin{cases} abku, & m = 0 \\ \dfrac{a[n(p\zeta - rl) - \eta ro]}{2}, & m = 1 \\ a[e(i\gamma - j\beta) + h(i\varepsilon - j\delta)], & m > 1 \end{cases}} \tag{11.41}$$

For $(\partial g_Z/\partial Z)_{n0}$, we have (from Eq. (11.37))

$$\left(\frac{\partial g_Z}{\partial Z}\right)_{n0} = \frac{\mu_\oplus}{R_\oplus^3}\left[\frac{(n+2)!}{n!}(C_{n0}V_{n+2,0} + S_{n0}W_{n+2,0})\right]$$

Noting again that $W_{n+2,0} = 0$,

$$\left(\frac{\partial g_Z}{\partial z}\right)_{n0} = \frac{\mu_\oplus}{R_\oplus^3}\left[\frac{(n+2)!}{n!}(C_{n0}V_{n+2,0})\right]$$

In terms of existing auxiliary parameters, we can rewrite this equation as

$$\left(\frac{\partial g_Z}{\partial z}\right)_{n0} = acko$$

For $(\partial g_Z/\partial Z)_{n1}$, we have (from Eq. (11.37))

$$\left(\frac{\partial g_Z}{\partial Z}\right)_{n1} = \frac{\mu_\oplus}{R_\oplus^3}\left[\frac{(n+1)!}{(n-1)!}(C_{n1}V_{n+2,1} + S_{n1}W_{n+2,1})\right]$$

In terms of existing auxiliary parameters, we can rewrite this equation as

$$\left(\frac{\partial g_Z}{\partial Z}\right)_{n1} = ad(pt + ru)$$

Using $(\partial g_Z/\partial Z)_{n0}$ and $(\partial g_Z/\partial Z)_{n1}$, we redefine $(\partial g_Z/\partial Z)_{nm}$ as

$$\boxed{\left(\frac{\partial g_Z}{\partial Z}\right)_{nm} = \begin{cases} acko, & m = 0 \\ ad(pt + ru), & m = 1 \\ af(ix + jy), & m > 1 \end{cases}} \tag{11.42}$$

At this point, we group the auxiliary parameters into three categories: one in which the parameter (a) is independent of both n and m, one in which the parameters are dependent only on n, and the third in which the parameters are dependent on both n and m.

$$\boxed{a = \frac{\mu_\oplus}{R_\oplus^3}} \tag{11.43}$$

$$\boxed{\begin{array}{llll} b = n+1 & c = (n+1)(n+2) & d = n(n+1) & k = C_{n0} \\ l = V_{n+2,2} & o = V_{n+2,0} & p = C_{n1} & q = V_{n+2,3} \\ r = S_{n1} & s = W_{n+2,3} & t = V_{n+2,1} & u = W_{n+2,1} \\ \zeta = W_{n+2,2} & \eta = n(n+1)(n+2) & & \end{array}} \tag{11.44}$$

$$\boxed{\begin{array}{llll} e = \dfrac{n-m+1}{2} & f = (m-n-2)(m-n-1) & g = \dfrac{(n-m+4)!}{(n-m)!} & h = \dfrac{(n-m+3)!}{2(n-m)!} \\ i = C_{nm} & j = S_{nm} & v = V_{n+2,m+2} & w = W_{n+2,m+2} \\ x = V_{n+2,m} & y = W_{n+2,m} & z = V_{n+2,m-2} & \alpha = W_{n+2,m-2} \\ \beta = V_{n+2,m+1} & \gamma = W_{n+2,m+1} & \delta = V_{n+2,m-1} & \varepsilon = W_{n+2,m-1} \end{array}} \tag{11.45}$$

Next, we want to assemble the full partial derivative matrix (i.e. Jacobian) in Eq. (11.33). However, we are missing the following four partial derivatives:

$$\frac{\partial g_Y}{\partial X}, \qquad \frac{\partial g_Y}{\partial Y}, \qquad \frac{\partial g_Z}{\partial X}, \qquad \text{and} \quad \frac{\partial g_Z}{\partial Y}$$

Recall that the gravitational accelerations are partial derivatives of the gravitational potential, $U(X, Y, Z)$. Then taking the partial derivative of g_Y with respect to X as an example, we have

$$\frac{\partial g_Y}{\partial X} = \frac{\partial}{\partial X}(g_Y) = \frac{\partial}{\partial X}\left(\frac{\partial U}{\partial Y}\right) = \frac{\partial^2 U}{\partial X \partial Y} = \frac{\partial^2 U}{\partial Y \partial X} = \frac{\partial}{\partial Y}\left(\frac{\partial U}{\partial X}\right) = \frac{\partial}{\partial Y}(g_X) = \frac{\partial g_X}{\partial Y}$$

It follows that

$$\boxed{\frac{\partial g_Y}{\partial X} = \frac{\partial g_X}{\partial Y}} \tag{11.46}$$

$$\boxed{\frac{\partial g_Z}{\partial X} = \frac{\partial g_X}{\partial Z}} \tag{11.47}$$

$$\boxed{\frac{\partial g_Z}{\partial Y} = \frac{\partial g_Y}{\partial Z}} \tag{11.48}$$

Finally, to find the last partial derivative, $\partial g_Y / \partial Y$, we can note that the sum of the diagonal elements of $[(\partial \mathbf{a}/\partial \mathbf{r})_g]_{\text{ecef}}$ (Eq. (11.33)) must be 0 [60, pp. 244–245]. Therefore,

$$\frac{\partial g_X}{\partial X} + \frac{\partial g_Y}{\partial Y} + \frac{\partial g_Z}{\partial Z} = 0 \quad \rightarrow \quad \boxed{\frac{\partial g_Y}{\partial Y} = -\frac{\partial g_X}{\partial X} - \frac{\partial g_Z}{\partial Z}} \tag{11.49}$$

The last step is to resolve $[(\partial \mathbf{a}/\partial \mathbf{r})_g]_{\text{ecef}}$ and $[(\partial \mathbf{a}/\partial \mathbf{v})_g]_{\text{ecef}}$ in the ECI frame. This can be done using a similarity transformation:

$$\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\text{eci}} = \left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\text{ecef}}^{-1}$$

$$\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_g\right]_{\text{eci}} = \left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_g\right]_{\text{ecef}}^{-1}$$

Immediately, we can see that

$$\boxed{\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_g\right]_{\text{eci}} = \mathbf{0}_{3\times 3}} \tag{11.50}$$

We can also note that $^{-1} = {}^T$ since rotation matrices are orthogonal. Therefore,

$$\boxed{\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\text{eci}} = \left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\text{ecef}}^T} \tag{11.51}$$

Algorithm 129: `dadX_gravity`

Partial derivatives of gravitational acceleration with respect to position and inertial velocity.

Inputs:

- $[\mathbf{r}]_{\text{ecef}} \in \mathbb{R}^3$ - position resolved in ECEF frame [m]
- $\mathbf{C} \in \mathbb{R}^{(N+1)\times(N+1)}$ - un-normalized gravitational coefficients
- $\mathbf{S} \in \mathbb{R}^{(N+1)\times(N+1)}$ - un-normalized gravitational coefficients
- $N \in \mathbb{R}$ - maximum degree/order of gravity model to use
- $\in \mathbb{R}^{3\times3}$ - rotation matrix from ECEF frame to ECI frame

Procedure:

1. Define $\mu_{\oplus}$ [m^3/s^2], and $R_{\oplus}$ [m] using the values from Appendix **??** (these values can be found in `GGM05S.txt` in the `toolbox/data/rawdata` folder).

2. Recursively compute the Legendre coefficients using Algorithm 124.

$$[\mathbf{V}, \mathbf{W}] = \texttt{legendre\_recursion}([\mathbf{r}]_{\text{ecef}}, R_{\oplus}, N)$$

3. Auxiliary parameter independent of n and m [s^{-2}].

$$a = \frac{\mu_{\oplus}}{R_{\oplus}^3}$$

4. Initialize the partial derivatives with respect to position [s^{-2}]

$$\frac{\partial g_X}{\partial X} = \frac{\partial g_X}{\partial Y} = \frac{\partial g_X}{\partial Z} = \frac{\partial g_Y}{\partial Z} = \frac{\partial g_Z}{\partial Z} = 0$$

5. Calculate the partial derivatives with respect to position.

   for $n = 0$ **to** N

     (a) Auxiliary parameters independent of m.

$$
\begin{aligned}
&b = n+1 &&c = (n+1)(n+2) &&d = n(n+1)\\
&k = C_{n0} &&l = V_{n+2,2} &&o = V_{n+2,0}\\
&p = C_{n1} &&q = V_{n+2,3} &&r = S_{n1}\\
&s = W_{n+2,3} &&t = V_{n+2,1} &&u = W_{n+2,1}\\
&\zeta = W_{n+2,2} &&\eta = n(n+1)(n+2)
\end{aligned}
$$

     (b) Zonal terms ($m = 0$) [s^{-2}].

$$
\begin{aligned}
\frac{\partial g_X}{\partial X} &= \frac{\partial g_X}{\partial X} + \frac{a(kl - cko)}{2}\\
\frac{\partial g_X}{\partial Y} &= \frac{\partial g_X}{\partial Y} + \frac{ak\zeta}{2}\\
\frac{\partial g_X}{\partial Z} &= \frac{\partial g_X}{\partial Z} + abkt\\
\frac{\partial g_Y}{\partial Z} &= \frac{\partial g_Y}{\partial Z} + abku\\
\frac{\partial g_Z}{\partial Z} &= \frac{\partial g_Z}{\partial Z} + acko
\end{aligned}
$$

(c) $m = 1$ terms $[\text{s}^{-2}]$.

$$\frac{\partial g_X}{\partial X} = \frac{\partial g_X}{\partial X} + \frac{a[pq + rs - d(3pt + ru)]}{4}$$

$$\frac{\partial g_X}{\partial Y} = \frac{\partial g_X}{\partial Y} + \frac{a[ps - rq - d(pu + rt)]}{4}$$

$$\frac{\partial g_X}{\partial Z} = \frac{\partial g_X}{\partial Z} + \frac{a[n(pl + r\zeta) - \eta po]}{2}$$

$$\frac{\partial g_Y}{\partial Z} = \frac{\partial g_Y}{\partial Z} + \frac{a[n(p\zeta - rl) - \eta ro]}{2}$$

$$\frac{\partial g_Z}{\partial Z} = \frac{\partial g_Z}{\partial Z} + ad(pt + ru)$$

(d) Remaining sectorial ($m = n$) and tesseral ($n > m$) terms $[\text{s}^{-2}]$.

for $m = 2$ **to** n

$$e = \frac{n - m + 1}{2} \qquad\qquad f = (m - n - 2)(m - n - 1)$$

$$g = (m - n - 4)(m - n - 3)f \quad h = \frac{(3 + n - m)f}{2}$$

$$i = C_{nm} \qquad\qquad j = S_{nm}$$

$$v = V_{n+2,m+2} \qquad\qquad w = W_{n+2,m+2}$$

$$x = V_{n+2,m} \qquad\qquad y = W_{n+2,m}$$

$$z = V_{n+2,m-2} \qquad\qquad \alpha = W_{n+2,m-2}$$

$$\beta = V_{n+2,m+1} \qquad\qquad \gamma = W_{n+2,m+1}$$

$$\delta = V_{n+2,m-1} \qquad\qquad \varepsilon = W_{n+2,m-1}$$

$$\frac{\partial g_X}{\partial X} = \frac{\partial g_X}{\partial X} + \frac{a[iv + jw - 2f(ix + jy) + g(iz + j\alpha)]}{4}$$

$$\frac{\partial g_X}{\partial Y} = \frac{\partial g_X}{\partial Y} + \frac{a[iw - jv + g(jz - i\alpha)]}{4}$$

$$\frac{\partial g_X}{\partial Z} = \frac{\partial g_X}{\partial Z} + a[e(i\beta + j\gamma) - h(i\delta + j\varepsilon)]$$

$$\frac{\partial g_Y}{\partial Z} = \frac{\partial g_Y}{\partial Z} + a[e(i\gamma - j\beta) + h(i\varepsilon - j\delta)]$$

$$\frac{\partial g_Z}{\partial Z} = \frac{\partial g_Z}{\partial Z} + af(ix + jy)$$

end
end

6. Remaining partial derivatives with respect to position, resolved in ECEF frame $[\text{s}^{-2}]$.

$$\frac{\partial g_Y}{\partial X} = \frac{\partial g_X}{\partial Y}$$

$$\frac{\partial g_Y}{\partial Y} = -\frac{\partial g_X}{\partial X} - \frac{\partial g_Z}{\partial Z}$$

$$\frac{\partial g_Z}{\partial X} = \frac{\partial g_X}{\partial Z}$$

$$\frac{\partial g_Z}{\partial Y} = \frac{\partial g_Y}{\partial Z}$$

7. Partial derivative of gravitational acceleration with respect to position, resolved in ECEF frame $[\mathrm{s}^{-2}]$.

$$
\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\mathrm{ecef}} = \begin{bmatrix} \dfrac{\partial g_X}{\partial X} & \dfrac{\partial g_X}{\partial Y} & \dfrac{\partial g_X}{\partial Z} \\[2mm] \dfrac{\partial g_Y}{\partial X} & \dfrac{\partial g_Y}{\partial Y} & \dfrac{\partial g_Y}{\partial Z} \\[2mm] \dfrac{\partial g_Z}{\partial X} & \dfrac{\partial g_Z}{\partial Y} & \dfrac{\partial g_Z}{\partial Z} \end{bmatrix}
$$

8. Partial derivative of gravitational acceleration with respect to position, resolved in ECI frame $[\mathrm{s}^{-2}]$.

$$
\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\mathrm{eci}} = \left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\mathrm{ecef}}^{T}
$$

9. Partial derivative of gravitational acceleration with respect to inertial velocity, resolved in ECI frame $[\mathrm{s}^{-1}]$.

$$
\left[\left(\frac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_g\right]_{\mathrm{eci}} = \mathbf{0}_{3\times 3}
$$

Outputs:

- $\left[\left(\dfrac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_g\right]_{\mathrm{eci}}$ - partial derivative of gravitational acceleration with respect to position, resolved in ECI frame $[\mathrm{s}^{-2}]$

- $\left[\left(\dfrac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_g\right]_{\mathrm{eci}}$ - partial derivative of gravitational acceleration with respect to inertial velocity, resolved in ECI frame $[\mathrm{s}^{-1}]$

11.7.1   Simplified Version: Point Mass Gravity

TODO rename to point mass For two-body (Kepler) gravity, we have

$$
\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_{\mathrm{kep}} = -\mu_\oplus \left(\frac{\mathbf{I}_{3\times 3}}{r^3} - \frac{3\mathbf{r}\mathbf{r}^T}{r^5}\right) = \frac{\mu_\oplus}{r^5} \begin{bmatrix} 3x^2 - r^2 & 3xy & 3xz \\ 3yx & 3y^2 - r^2 & 3yz \\ 3zx & 3zy & 3z^2 - r^2 \end{bmatrix} \tag{11.52}
$$

$$
\left(\frac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_{\mathrm{kep}} = \mathbf{0}_{3\times 3} \tag{11.53}
$$

These expressions are independent of the coordinate frame. It is assumed that for an arbitrary geocentric coordinate frame E, $[\mathbf{r}]_E = (x, y, z)^T$ [60, p. 244].

Algorithm 130: `dadX_two_body`

Partial derivatives of two-body/Keplerian gravitational acceleration with respect to position and inertial velocity.

Inputs:

- $\mathbf{r} \in \mathbb{R}^3$ - position $[\mathrm{m}]$

Procedure:
1. Define $\mu_\oplus$ using the value from Appendix **??**.
2. Position vector magnitude [m].

$$r = \|\mathbf{r}\|$$

3. Partial derivative of two-body/Keplerian gravitational acceleration with respect to position [s^{-2}].

$$\left(\frac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_{\text{kep}} = -\mu_\oplus \left(\frac{\mathbf{I}_{3\times 3}}{r^3} - \frac{3\mathbf{r}\mathbf{r}^T}{r^5}\right)$$

4. Partial derivative of two-body/Keplerian gravitational acceleration with respect to inertial velocity [s^{-1}].

$$\left(\frac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_{\text{kep}} = \mathbf{0}_{3\times 3}$$

Outputs:
- $\left(\dfrac{\partial \mathbf{a}}{\partial \mathbf{r}}\right)_{\text{kep}}$ - partial derivative of two-body/Keplerian gravitational acceleration with respect to position [s^{-2}]
- $\left(\dfrac{\partial \mathbf{a}}{\partial \mathbf{v}}\right)_{\text{kep}}$ - partial derivative of two-body/Keplerian gravitational acceleration with respect to inertial velocity [s^{-1}]

Note:
- $\mathbf{r}$ must be defined in a geocentric coordinate frame.

11.8 Third Body Gravity

Let's consider the general case when there are N bodies, and we are interested in the acceleration of body 2 with respect to body 1. Then we have (from Eq. (**??**))

$$\ddot{\mathbf{r}}_{12} = -G(m_1 + m_2)\frac{\mathbf{r}_{12}}{r_{12}{}^3} - G\sum_{k=3}^{N} m_k \left(\frac{\mathbf{r}_{k2}}{r_{k2}{}^3} - \frac{\mathbf{r}_{k1}}{r_{k1}{}^3}\right)$$

If we restrict ourselves to $N = 3$ bodies,

$$\ddot{\mathbf{r}}_{12} = -G(m_1 + m_2)\frac{\mathbf{r}_{12}}{r_{12}{}^3} - Gm_3 \left(\frac{\mathbf{r}_{32}}{r_{32}{}^3} - \frac{\mathbf{r}_{31}}{r_{31}{}^3}\right)$$

Switching the sign on the second term,

$$\ddot{\mathbf{r}}_{12} = -G(m_1 + m_2)\frac{\mathbf{r}_{12}}{r_{12}{}^3} + Gm_3 \left(-\frac{\mathbf{r}_{32}}{r_{32}{}^3} + \frac{\mathbf{r}_{31}}{r_{31}{}^3}\right)$$

Noting that $\mathbf{r}_{32} = -\mathbf{r}_{23}$ and $\mathbf{r}_{31} = -\mathbf{r}_{13}$,

$$\ddot{\mathbf{r}}_{12} = -G(m_1 + m_2)\frac{\mathbf{r}_{12}}{r_{12}{}^3} + Gm_3 \left(\frac{\mathbf{r}_{23}}{r_{23}{}^3} - \frac{\mathbf{r}_{13}}{r_{13}{}^3}\right)$$

Recall from Section **??** that we took body 1 to be the Earth and body 2 to be the satellite. Additionally, we assumed that $m_1 \gg m_2$ such that $G(m_1 + m_2) \approx Gm_1 = \mu_\oplus$, and we let $\mathbf{r} = \mathbf{r}_{12}$ since the position of the satellite

is taken to be the position of the satellite with respect to the Earth. Additionally, note that $Gm_3 = \mu_3$. Thus,

$$\ddot{\mathbf{r}} = -\mu_\oplus \frac{\mathbf{r}}{r^3} + \mu_3 \left(\frac{\mathbf{r}_{\text{sat}3}}{r_{\text{sat}3}^3} - \frac{\mathbf{r}_{\oplus 3}}{r_{\oplus 3}^3} \right)$$

We know that

$$\mathbf{r}_{\text{sat}3} = \mathbf{r}_{\text{sat}} - \mathbf{r}_3 = \mathbf{r} - \mathbf{r}_3$$

which implies that

$$r_{\text{sat}3} = \|\mathbf{r} - \mathbf{r}_3\|^3$$

We also know that

$$\mathbf{r}_{\oplus 3} = \mathbf{r}_3$$

since we express these vectors in a geocentric reference frame (i.e. $\mathbf{r}_\oplus = \mathbf{0}$ since the origin is located at the center of the Earth). Thus, we arrive at

$$\ddot{\mathbf{r}} = \underbrace{-\mu_\oplus \frac{\mathbf{r}}{r^3}}_{\substack{\text{2-body} \\ \text{acceleration}}} + \underbrace{\mu_3 \left(\frac{\mathbf{r}_3 - \mathbf{r}}{\|\mathbf{r}_3 - \mathbf{r}\|^3} - \frac{\mathbf{r}_3}{\|\mathbf{r}_3\|^3} \right)}_{\text{3rd-body perturbation}}$$

At this point, we can identify the third-body perturbing acceleration as [60, p. 69][103, p. 574]

$$\boxed{\mathbf{f}_3 = \mu_3 \left(\frac{\mathbf{r}_3 - \mathbf{r}}{\|\mathbf{r}_3 - \mathbf{r}\|^3} - \frac{\mathbf{r}_3}{\|\mathbf{r}_3\|^3} \right)} \tag{11.54}$$

This equation for the third-body perturbation can be used for Earth satellites with no noticeable numerical difficulties. However, for orbits around other central bodies (such as the Moon), this form can pose significant numerical problems because $\mathbf{r}_3 - \mathbf{r} \approx \mathbf{r}_3$. A discussion is included in [103, p. 575] to resolve this issue.

Below, we introduce Algorithm 131 to determine the perturbing acceleration due to the gravity of a third body, such as the Sun or the Moon[17].

Algorithm 131: `third_body`
Perturbing acceleration due to third-body gravity.

Inputs:
- $\mathbf{r} \in \mathbb{R}^3$ - satellite position [m]
- $\mathbf{r}_3 \in \mathbb{R}^3$ - third body position [m]
- $\mu_3 \in \mathbb{R}$ - third body gravitational parameter [m³/s²]

Procedure:

$$\mathbf{f}_3 = \mu_3 \left(\frac{\mathbf{r}_3 - \mathbf{r}}{\|\mathbf{r}_3 - \mathbf{r}\|^3} - \frac{\mathbf{r}_3}{\|\mathbf{r}_3\|^3} \right)$$

Outputs:
- $\mathbf{f}_3 \in \mathbb{R}^3$ - perturbing acceleration due to third-body gravity [m/s²]

Note:
- $\mathbf{r}$ and $\mathbf{r}_3$ must be resolved in the same geocentric coordinate frame.

---

[17] For these two bodies, we can use values from Appendix **??** for their standard gravitational parameters, and Algorithms 133 and 134 (in Sections 12.2.1 and 12.2.2, respectively) for lower-fidelity models of their positions.

12

Space Environment

TODO: update all tikz diagrams in separate latex doc

12.1 Obliquity of the Ecliptic

[103, p. 216] gives the **mean obliquity of the ecliptic**, $\bar{\varepsilon}$, in degrees as

$$
\boxed{
\begin{aligned}
(\bar{\varepsilon})^{\circ} &= 23.439279 - 0.0130102 T_{\mathrm{TT}} - (5.086 \times 10^{-8}) T_{\mathrm{TT}}^2 + (5.565 \times 10^{-7}) T_{\mathrm{TT}}^3 + (1.6 \times 10^{-10}) T_{\mathrm{TT}}^4 \\
&\quad + (1.25 \times 10^{-11}) T_{\mathrm{TT}}^5
\end{aligned}
}
\tag{12.1}
$$

Here, we *approximate* the **true obliquity of the ecliptic**, ε, using the mean obliquity (i.e. we assume $\varepsilon \approx \bar{\varepsilon}$). Algorithm 132 implements the calculation of the obliquity as a function of the modified Julian date of TT.

Algorithm 132: `obliquity`
Obliquity of the Ecliptic.

Inputs:
- $\mathrm{MJD}_{\mathrm{TT}} \in \mathbb{R}$ - TT (Terrestrial Time) $[\mathrm{MJD}]$

Procedure:
1. Julian centuries since J2000.0 of TT $[\mathrm{c}]$ (Algorithm 70).

$$
T_{\mathrm{TT}} = \texttt{mjd2t}(\mathrm{MJD}_{\mathrm{TT}})
$$

2. Obliquity $[^{\circ}]$.

$$
\begin{aligned}
\varepsilon^{\circ} &\approx 23.439279 - 0.0130102 T_{\mathrm{TT}} - (5.086 \times 10^{-8}) T_{\mathrm{TT}}^2 \\
&\quad + (5.565 \times 10^{-7}) T_{\mathrm{TT}}^3 + (1.6 \times 10^{-10}) T_{\mathrm{TT}}^4 \\
&\quad + (1.25 \times 10^{-11}) T_{\mathrm{TT}}^5
\end{aligned}
$$

3. Convert obliquity to radians (Algorithm 88).

$$
\varepsilon = \texttt{deg2rad}(\varepsilon^{\circ})
$$

Outputs:
- $\varepsilon \in \mathbb{R}$ - obliquity of the ecliptic [rad]

Note:
- This algorithm assumes the true obliquity is approximately equal to the mean obliquity (i.e. $\varepsilon \approx \bar{\varepsilon}$).

12.2 Positions of Celestial Bodies

12.2.1 Moon Position

Algorithm 133 below is adapted from Algorithm 31 in [103, p. 288]

Algorithm 133: `moon_position`

Moon position vector resolved in the ECI frame.

Inputs:
- $\mathrm{MJD}_{\mathrm{UT1}} \in \mathbb{R}$ - UT1 (Universal Time 1) [MJD]
- $\varepsilon \in \mathbb{R}$ - obliquity of the ecliptic [rad]

Note:
- This algorithm assumes that $\mathrm{TDB} \approx \mathrm{UT1}$.

Procedure:

1. Define $R_\oplus$ using the value from Appendix **??**.
2. Julian centuries since J2000.0 of UT1 [c] (Algorithm 70).

$$T_{\mathrm{UT1}} = \texttt{mjd2t}(\mathrm{MJD}_{\mathrm{UT1}})$$

3. Ecliptic longitude [°] of the Moon.

$$\begin{aligned}
\lambda_\leftmoon =\ & 218.32 + 481267.8813 T_{\mathrm{UT1}} \\
& + 6.29 \sin\left(134.9 + 477198.85 T_{\mathrm{UT1}}\right) \\
& - 1.27 \sin\left(259.2 - 413335.38 T_{\mathrm{UT1}}\right) \\
& + 0.66 \sin\left(235.7 + 890534.23 T_{\mathrm{UT1}}\right) \\
& + 0.21 \sin\left(269.9 + 954397.70 T_{\mathrm{UT1}}\right) \\
& - 0.19 \sin\left(357.5 + 35999.05 T_{\mathrm{UT1}}\right) \\
& - 0.11 \sin\left(186.6 + 966404.05 T_{\mathrm{UT1}}\right) \\
\lambda_\leftmoon =\ & \mathrm{mod}\left(\lambda_\leftmoon, 360°\right)
\end{aligned}$$

4. Ecliptic latitude [°] of the Moon.

$$\begin{aligned}
\phi_\leftmoon =\ & 5.13 \sin\left(93.3 + 483202.03 T_{\mathrm{UT1}}\right) \\
& + 0.28 \sin\left(228.2 + 960400.87 T_{\mathrm{UT1}}\right) \\
& - 0.28 \sin\left(318.3 + 6003.18 T_{\mathrm{UT1}}\right) \\
& - 0.17 \sin\left(217.6 - 407332.20 T_{\mathrm{UT1}}\right) \\
\phi_\leftmoon =\ & \mathrm{mod}\left(\phi_\leftmoon, 360\right)
\end{aligned}$$

5. Horizontal parallax [°].

$$\begin{aligned}
\mathcal{P} = \ & 0.9508 + 0.0518\cos\left(134.9 + 477198.85 T_{\mathrm{UT1}}\right) \\
& + 0.0095\cos\left(259.2 - 413335.38 T_{\mathrm{UT1}}\right) \\
& + 0.0078\cos\left(235.7 + 890534.23 T_{\mathrm{UT1}}\right) \\
& + 0.0028\cos\left(269.9 + 954397.70 T_{\mathrm{UT1}}\right)
\end{aligned}$$

6. Earth mean equatorial radius.

$$R_\oplus = 6378136.3 \text{ m}$$

7. Distance from Earth to Moon [m].

$$r_\mathbb{C} = \frac{R_\oplus}{\sin\mathcal{P}}$$

8. Converts the obliquity of the ecliptic to degrees (Algorithm 89).

$$\varepsilon = \texttt{rad2deg}(\varepsilon)$$

9. Moon position resolved in ECI frame [m].

$$[\mathbf{r}_\mathbb{C}]_{\mathrm{eci}} = r_\mathbb{C} \begin{bmatrix} \cos\left(\phi_\mathbb{C}\right)\cos\left(\lambda_\mathbb{C}\right) \\ \cos\left(\varepsilon\right)\cos\left(\phi_\mathbb{C}\right)\sin\left(\lambda_\mathbb{C}\right) - \sin\left(\varepsilon\right)\sin\left(\phi_\mathbb{C}\right) \\ \sin\left(\varepsilon\right)\cos\left(\phi_\mathbb{C}\right)\sin\left(\lambda_\mathbb{C}\right) + \cos\left(\varepsilon\right)\sin\left(\phi_\mathbb{C}\right) \end{bmatrix}$$

Outputs:
- $[\mathbf{r}_\mathbb{C}]_{\mathrm{eci}} \in \mathbb{R}^3$ - Moon position resolved in ECI frame [m]

12.2.2 Sun Position

Algorithm 134 below is adapted from Algorithm 29 in [103, pp. 279–280]

Algorithm 134: `sun_position`
Sun position vector resolved in the ECI frame.

Inputs:
- $\mathrm{MJD}_{\mathrm{UT1}} \in \mathbb{R}$ - UT1 (Universal Time 1) [MJD]
- $\mathrm{MJD}_{\mathrm{TT}} \in \mathbb{R}$ - TT (Terrestrial Time) [MJD]

Procedure:
1. Julian centuries since J2000.0 of UT1 and TT [c] (Algorithm 70).

$$\begin{aligned}
T_{\mathrm{UT1}} &= \texttt{mjd2t}(\mathrm{MJD}_{\mathrm{UT1}}) \\
T_{\mathrm{TT}} &= \texttt{mjd2t}(\mathrm{MJD}_{\mathrm{TT}})
\end{aligned}$$

2. Mean ecliptic longitude [°] and mean anomaly of the Sun (in the ECI frame) [°].

$$\begin{aligned}
\lambda_{M,\odot} &= 280.46 + 36000.771 T_{\mathrm{UT1}} \\
M_\odot &= 357.5291092 + 35999.05034 T_{\mathrm{UT1}}
\end{aligned}$$

3. Ecliptic longitude of the Sun [°].

$$\lambda_\odot = \lambda_{M,\odot} + 1.914666471 \sin M_\odot + 0.019994643 \sin \left(2M_\odot \right)$$

4. Distance from Earth to Sun [AU].

$$r_\odot = 1.000140612 - 0.016708617 \cos M_\odot - 0.000139589 \cos \left(2M_\odot \right)$$

5. Converts $r_\odot$ from AU to m.

$$r_\odot = (149597870000) r_\odot$$

6. Obliquity of the ecliptic [rad] (Algorithm 132).

$$\varepsilon = \texttt{obliquity}(T_{\mathrm{TT}})$$

7. Convert obliquity to degrees (Algorithm 89).

$$\varepsilon = \texttt{rad2deg}(\varepsilon)$$

8. Sun position resolved in ECI frame [m].

$$[\mathbf{r}_\odot]_{\mathrm{eci}} = r_\odot \begin{bmatrix} \cos \lambda_\odot \\ \sin \lambda_\odot \cos \varepsilon \\ \sin \lambda_\odot \sin \varepsilon \end{bmatrix}$$

Outputs:
- $[\mathbf{r}_\odot]_{\mathrm{eci}} \in \mathbb{R}^3$ - Sun position resolved in ECI frame [m]

12.3   Eclipses

A satellite is in **eclipse** if the Earth blocks its view of the Sun. Clearly, to determine if a satellite is in eclipse, we first need to find the position of the Sun, which we can do using Algorithm 134.

12.3.1   Cylindrical Eclipse Model

TODO: finish this up TODO: unit tests in latex TODO: unit tests in matlab TODO: update docs

To find out when the satellite is in eclipse, we can begin by considering the cylindrical shadow model shown in Fig. 12.1, where the 2D diagram is assumed to be in the orbital plane of the satellite. By inspection of the geometry, we can see that there are two conditions that are necessary and sufficient to determine when an eclipse will occur:

> **Eclipse Conditions (Cylindrical Eclipse Model)**
>   1. $\|\boldsymbol{r}_\perp\| < R_\oplus$
>   2. $\boldsymbol{r}_\odot \cdot \boldsymbol{r} < 0$

From Fig. 12.1, we can see that

$$\boldsymbol{r} = \boldsymbol{r}_\perp + \boldsymbol{r}_\parallel \tag{12.2}$$

Additionally, we know $\boldsymbol{r}_\parallel$ is the vector projection of $\boldsymbol{r}$ onto $\boldsymbol{r}_\odot$ (where $\hat{\boldsymbol{r}}_\odot = \boldsymbol{r}_\odot / r_\odot$).

$$\boxed{\boldsymbol{r}_\parallel = (\boldsymbol{r} \cdot \hat{\boldsymbol{r}}_\odot) \hat{\boldsymbol{r}}_\odot} \tag{12.3}$$
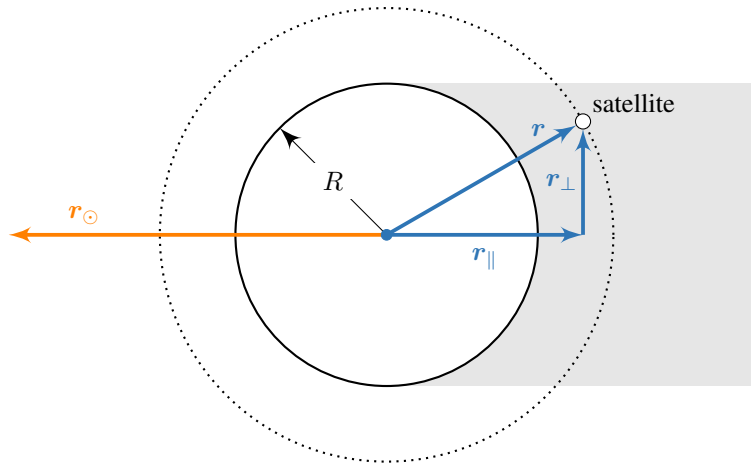
Figure 12.1: Cylindrical shadow model.

Knowing $r_\parallel$, we can rearrange Eq. (12.2) to solve for $r_\perp$.

$$\boxed{r_\perp = r - r_\parallel}$$
(12.4)

Using the above equations and conditions for eclipse, we define Algorithm 135 to determine whether a satellite is in eclipse or not. Note that the equations above don't depend on what coordinate frame the vectors are expressed in, as long as they are both expressed in the same coordinate frame. We will usually have the Sun position vector expressed in the ECI frame, so it is convenient to use the satellite position vector expressed in that frame as well [**damico_279c_lecture_10**].

Algorithm 135: `eclipse`
Determines if a satellite is in eclipse (cylindrical shadow model).

Inputs:
- $r \in \mathbb{R}^3$ - satellite position [m]
- $r_\odot \in \mathbb{R}^3$ - Sun position [m]

Procedure:
1. Define $R_\oplus$ [m] using the value from Appendix **??**.
2. Sun unit position vector (pointing from center of Earth towards Sun).

$$\hat{r}_\odot = \frac{r_\odot}{\|r_\odot\|}$$

3. Dot product of satellite position with the Sun position unit vector.

$$d = r \cdot \hat{r}_\odot$$

4. Parallel/perpendicular decomposition of r.

$$r_\parallel = d\hat{r}_\odot$$
$$r_\perp = r - r_\parallel$$

5. Determines if satellite is in eclipse. *Note that $r_\odot \cdot r = r \cdot r_\odot$, which has the same sign as $r \cdot \hat{r}_\odot = d$.*

$$\texttt{in\_eclipse} = [(\|r_\perp\| < R_\oplus) \text{ and } (d < 0)]$$

Outputs:

- `in_eclipse` $\in \mathbb{B}$ - `true` if satellite is in eclipse, `false` otherwise

Note:

- r and $r_{\odot}$ must be expressed in the same Earth-centered coordinate frame.

12.3.2 Conical Eclipse Model

TODO

   TODO: eclipse entrance and exit times

12.4 Solar Radiation

The mean integrated solar energy flux at the Earth's position is given by

$$F_e = \frac{1358}{1.0004 + 0.0334 \cos D} \text{ W/m}^2$$

where D is 2π times the number of days since July 4 (the approximate aphelion). 1358 W/m^2 is the mean flux at 1 AU, so the denominator is a correction for the true Earth distance [103, p. 579][105, p. 130]. Typically, F_e is approximated as [103, p. 578]

$$F_e \approx 1367 \text{ W/m}^2$$

The solar radiation pressure is then defined as [103, p. 580]

$$P_{\odot} = \frac{F_e}{c} = \frac{1367 \text{ W/m}^2}{299792458 \text{ m/s}} = 4.559821... \times 10^{-6} \text{ N/m}^2$$

We typically just keep the first three significant digits and use [60, p. 77]

$$\boxed{P_{\odot} = 4.56 \times 10^{-6} \text{ N/m}^2}$$

(12.5)

PART IV

Appendices

A

Test Cases

A.1 Rotation Test Cases

A.1.1 Rotation Matrices

Elementary Rotations

θ [rad]	$\mathbf{R}_1(\theta)$ rot1 (Algorithm 1)	$\mathbf{R}_2(\theta)$ rot2 (Algorithm 2)	$\mathbf{R}_3(\theta)$ rot3 (Algorithm 3)
0	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\pi/6$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \sqrt{3}/2 & 1/2 \\ 0 & -1/2 & \sqrt{3}/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{3}/2 & 0 & -1/2 \\ 0 & 1 & 0 \\ 1/2 & 0 & \sqrt{3}/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{3}/2 & 1/2 & 0 \\ -1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\pi/4$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \sqrt{2}/2 & \sqrt{2}/2 \\ 0 & -\sqrt{2}/2 & \sqrt{2}/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{2}/2 & 0 & -\sqrt{2}/2 \\ 0 & 1 & 0 \\ \sqrt{2}/2 & 0 & \sqrt{2}/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ -\sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\pi/3$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & \sqrt{3}/2 \\ 0 & -\sqrt{3}/2 & 1/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{1}/2 & 0 & -\sqrt{3}/2 \\ 0 & 1 & 0 \\ \sqrt{3}/2 & 0 & 1/2 \end{bmatrix}$	$\begin{bmatrix} 1/2 & \sqrt{3}/2 & 0 \\ -\sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\pi/2$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$2\pi/3$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1/2 & \sqrt{3}/2 \\ 0 & -\sqrt{3}/2 & -1/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{1}/2 & 0 & \sqrt{3}/2 \\ 0 & 1 & 0 \\ -\sqrt{3}/2 & 0 & -1/2 \end{bmatrix}$	$\begin{bmatrix} 1/2 & \sqrt{3}/2 & 0 \\ -\sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$3\pi/4$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -\sqrt{2}/2 & \sqrt{2}/2 \\ 0 & -\sqrt{2}/2 & -\sqrt{2}/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{2}/2 & 0 & -\sqrt{2}/2 \\ 0 & 1 & 0 \\ \sqrt{2}/2 & 0 & -\sqrt{2}/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{2}/2 & \sqrt{2}/2 & 0 \\ -\sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$5\pi/6$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -\sqrt{3}/2 & 1/2 \\ 0 & -1/2 & -\sqrt{3}/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{3}/2 & 0 & -1/2 \\ 0 & 1 & 0 \\ 1/2 & 0 & -\sqrt{3}/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{3}/2 & 1/2 & 0 \\ -1/2 & -\sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
π	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$7\pi/6$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -\sqrt{3}/2 & -1/2 \\ 0 & 1/2 & -\sqrt{3}/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{3}/2 & 0 & 1/2 \\ 0 & 1 & 0 \\ -1/2 & 0 & -\sqrt{3}/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{3}/2 & -1/2 & 0 \\ 1/2 & -\sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$5\pi/4$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -\sqrt{2}/2 & -\sqrt{2}/2 \\ 0 & \sqrt{2}/2 & -\sqrt{2}/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{2}/2 & 0 & \sqrt{2}/2 \\ 0 & 1 & 0 \\ -\sqrt{2}/2 & 0 & -\sqrt{2}/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$4\pi/3$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1/2 & -\sqrt{3}/2 \\ 0 & \sqrt{3}/2 & -1/2 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{1}/2 & 0 & \sqrt{3}/2 \\ 0 & 1 & 0 \\ -\sqrt{3}/2 & 0 & -1/2 \end{bmatrix}$	$\begin{bmatrix} -1/2 & -\sqrt{3}/2 & 0 \\ \sqrt{3}/2 & -1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$3\pi/2$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$5\pi/3$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & -\sqrt{3}/2 \\ 0 & \sqrt{3}/2 & 1/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{1}/2 & 0 & \sqrt{3}/2 \\ 0 & 1 & 0 \\ -\sqrt{3}/2 & 0 & 1/2 \end{bmatrix}$	$\begin{bmatrix} 1/2 & -\sqrt{3}/2 & 0 \\ \sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$7\pi/4$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \sqrt{2}/2 & -\sqrt{2}/2 \\ 0 & \sqrt{2}/2 & \sqrt{2}/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{2}/2 & 0 & \sqrt{2}/2 \\ 0 & 1 & 0 \\ -\sqrt{2}/2 & 0 & \sqrt{2}/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$11\pi/6$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \sqrt{3}/2 & -1/2 \\ 0 & 1/2 & \sqrt{3}/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{3}/2 & 0 & 1/2 \\ 0 & 1 & 0 \\ -1/2 & 0 & \sqrt{3}/2 \end{bmatrix}$	$\begin{bmatrix} \sqrt{3}/2 & -1/2 & 0 \\ 1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
2π	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

3-2-1 Rotation Sequence (`rot321`)

"Forward" Test:

$$\theta_1 = 30 \left(\frac{\pi}{180} \right)$$
$$\theta_2 = -40 \left(\frac{\pi}{180} \right)$$
$$\theta_3 = 50 \left(\frac{\pi}{180} \right)$$

$\texttt{rot321}(\theta_1, \theta_2, \theta_3)$ should equal $\texttt{rot1}(\theta_3)\texttt{rot2}(\theta_2)\texttt{rot3}(\theta_1)$

"Reverse" Test:

$$\theta_1 = 30\left(\frac{\pi}{180}\right)$$

$$\theta_2 = -40\left(\frac{\pi}{180}\right)$$

$$\theta_3 = 50\left(\frac{\pi}{180}\right)$$

$$\texttt{rot321}(\theta_1, \theta_2, \theta_3)^T \qquad \text{should equal} \qquad \texttt{rot3}(-\theta_1)\texttt{rot2}(-\theta_2)\texttt{rot3}(-\theta_3)$$

3-1-3 Rotation Sequence (`rot313`)

"Forward" Test:

$$\theta_1 = 30\left(\frac{\pi}{180}\right)$$

$$\theta_2 = -40\left(\frac{\pi}{180}\right)$$

$$\theta_3 = 50\left(\frac{\pi}{180}\right)$$

$$\texttt{rot313}(\theta_1, \theta_2, \theta_3) \qquad \text{should equal} \qquad \texttt{rot3}(\theta_3)\texttt{rot1}(\theta_2)\texttt{rot3}(\theta_1)$$

"Reverse" Test:

$$\theta_1 = 30\left(\frac{\pi}{180}\right)$$

$$\theta_2 = -40\left(\frac{\pi}{180}\right)$$

$$\theta_3 = 50\left(\frac{\pi}{180}\right)$$

$$\texttt{rot313}(\theta_1, \theta_2, \theta_3)^T \qquad \text{should equal} \qquad \texttt{rot3}(-\theta_1)\texttt{rot1}(-\theta_2)\texttt{rot3}(-\theta_3)$$

`matrotate` (Algorithm 8)

$$\mathbf{R}_{A \to B} = \begin{bmatrix} 0.5721 & 0.4156 & -0.7071 \\ -0.7893 & 0.0446 & -0.6124 \\ -0.2230 & 0.9084 & 0.3536 \end{bmatrix}, \quad [r]_A = \begin{bmatrix} 5 \\ 4 \\ 3 \end{bmatrix}$$

$$[r]_B = \texttt{matrotate}(\mathbf{R}_{A \to B}, [r]_A) = \begin{bmatrix} 2.4016 \\ -5.6053 \\ 3.5794 \end{bmatrix}$$

`matchain` (Algorithm 9)

$$\mathbf{R}_{A \to B} = \begin{bmatrix} 0.5721 & 0.4156 & -0.7071 \\ -0.7893 & 0.0446 & -0.6124 \\ -0.2230 & 0.9084 & 0.3536 \end{bmatrix}, \quad \mathbf{R}_{B \to C} = \begin{bmatrix} -0.5721 & -0.5721 & 0.5878 \\ 0.0064 & 0.7135 & 0.7006 \\ -0.8202 & 0.4046 & -0.4045 \end{bmatrix}$$

$$\mathbf{R}_{A \to C} = \texttt{matchain}(\mathbf{R}_{A \to B}, \mathbf{R}_{B \to C}) = \begin{bmatrix} -0.0068 & 0.2707 & 0.9627 \\ -0.7157 & 0.6709 & -0.1937 \\ -0.6984 & -0.6903 & 0.1892 \end{bmatrix}$$

A.1.2 Conversions Between Rotation Parameterizations

To test the conversions between the different rotation parameterizations, we take a systematic approach; once we verify one algorithm using some unit tests, we can use that algorithm when constructing unit tests to verify other algorithms.

eul2mat_321 (Algorithm 26)

Note that `eul2mat_321` (Algorithm 26) is just a wrapper around `rot321` (Algorithm 4), where we set $\theta_1 = \psi$, $\theta_2 = \theta$, and $\theta_3 = \phi$. To simply check that we performed the wrapping correctly, we can perform the following test:

$$\psi = 30 \left(\frac{\pi}{180} \right)$$
$$\theta = -40 \left(\frac{\pi}{180} \right)$$
$$\phi = 50 \left(\frac{\pi}{180} \right)$$
$$\texttt{eul2mat\_321}(\psi, \theta, \phi) \quad \Longrightarrow \quad \texttt{rot321}(\psi, \theta, \phi)$$

As an additional check, we can perform the following numerical comparison:

$$\psi = \frac{\pi}{6}$$
$$\theta = -\frac{\pi}{6}$$
$$\phi = \frac{3\pi}{4}$$
$$\texttt{eul2mat\_321}(\psi, \theta, \phi) \quad \Longrightarrow \quad \begin{bmatrix} -0.6124 & 0.6124 & 0.5000 \\ -0.4356 & -0.7891 & 0.4330 \\ 0.6597 & 0.0474 & 0.7500 \end{bmatrix}$$

mat2eul_321 (Algorithm 27)

After `eul2mat_321` has been unit tested, we can unit test `mat2eul_321` (Algorithm 27) by defining a set of 3-2-1 Euler angles, constructing a rotation matrix from those angles using `eul2mat_321`, and then checking whether `mat2eul_321` correctly recovers those angles.

$$\psi = \frac{3\pi}{4}$$
$$\theta = -\frac{\pi}{6}$$
$$\phi = \frac{\pi}{6}$$
$$\mathbf{R} = \texttt{eul2mat\_321}(\psi, \theta, \phi)$$
$$\texttt{mat2eul\_321}(\mathbf{R}) \quad \Longrightarrow \quad \psi = \frac{\pi}{6}, \quad \theta = -\frac{\pi}{6}, \quad \phi = \frac{3\pi}{4}$$

Numerically,

$$\mathbf{R} = \begin{bmatrix} -0.6124 & 0.6124 & 0.5000 \\ -0.4356 & -0.7891 & 0.4330 \\ 0.6597 & 0.0474 & 0.7500 \end{bmatrix} \quad \Longrightarrow \quad \psi = \frac{\pi}{6}, \quad \theta = -\frac{\pi}{6}, \quad \phi = \frac{3\pi}{4}$$

Next, we can test the pitch-up singularity using a yaw angle of 0, in which case the rotation matrix to 3-2-1 Euler angle

conversion should correctly recover the roll angle.

$$\psi = 0$$
$$\theta = \frac{\pi}{2}$$
$$\phi = \frac{\pi}{5}$$
$$\mathbf{R} = \texttt{eul2mat\_321}(\psi, \theta, \phi)$$
$$\texttt{mat2eul\_321}(\mathbf{R}) \quad \implies \quad \psi = 0, \quad \theta = \frac{\pi}{2}, \quad \phi = \frac{\pi}{5}$$

Testing the pitch-up singularity with a yaw angle of $-\pi/6$,

$$\psi = -\frac{\pi}{6}$$
$$\theta = \frac{\pi}{2}$$
$$\phi = \frac{\pi}{5}$$
$$\mathbf{R} = \texttt{eul2mat\_321}(\psi, \theta, \phi)$$
$$\texttt{mat2eul\_321}(\mathbf{R}) \quad \implies \quad \psi = 0, \quad \theta = \frac{\pi}{2}, \quad \phi = 1.1519$$

Next, we can test the pitch-down singularity using a yaw angle of 0, in which case the rotation matrix to 3-2-1 Euler angle conversion should correctly recover the roll angle.

$$\psi = 0$$
$$\theta = -\frac{\pi}{2}$$
$$\phi = \frac{\pi}{5}$$
$$\mathbf{R} = \texttt{eul2mat\_321}(\psi, \theta, \phi)$$
$$\texttt{mat2eul\_321}(\mathbf{R}) \quad \implies \quad \psi = 0, \quad \theta = -\frac{\pi}{2}, \quad \phi = \frac{\pi}{5}$$

Testing the pitch-down singularity with a yaw angle of $-\pi/6$,

$$\psi = -\frac{\pi}{6}$$
$$\theta = -\frac{\pi}{2}$$
$$\phi = \frac{\pi}{5}$$
$$\mathbf{R} = \texttt{eul2mat\_321}(\psi, \theta, \phi)$$
$$\texttt{mat2eul\_321}(\mathbf{R}) \quad \implies \quad \psi = 0, \quad \theta = -\frac{\pi}{2}, \quad \phi = 0.1047$$

Next, we need to check the cases where the R_{13} element of the rotation matrix is slightly less than -1 or slightly greater than 1 due to numerical issues near the pitch-up and pitch-down singularities. For the pitch-up case, we construct a rotation matrix with a pitch-up singularity as before, subtract 10^{-14} to the R_{13} element (which should originally be

$R_{13} = -1$), and then convert it back to 3-2-1 Euler angles.

$$\psi = -\frac{\pi}{6}$$

$$\theta = \frac{\pi}{2}$$

$$\phi = \frac{\pi}{5}$$

$$\mathbf{R} = \texttt{eul2mat\_321}(\psi, \theta, \phi)$$

$$R_{13} = R_{13} - 10^{-14}$$

$$\texttt{mat2eul\_321}(\mathbf{R}) \qquad \Longrightarrow \qquad \psi = 0, \quad \theta = \frac{\pi}{2}, \quad \phi = 1.1519$$

Next, we do the same for the pitch-down case, but instead *add* 10^{-14} to the R_{13} element (which should originally be $R_{13} = 1$).

$$\psi = -\frac{\pi}{6}$$

$$\theta = -\frac{\pi}{2}$$

$$\phi = \frac{\pi}{5}$$

$$\mathbf{R} = \texttt{eul2mat\_321}(\psi, \theta, \phi)$$

$$R_{13} = R_{13} + 10^{-14}$$

$$\texttt{mat2eul\_321}(\mathbf{R}) \qquad \Longrightarrow \qquad \psi = 0, \quad \theta = -\frac{\pi}{2}, \quad \phi = 0.1047$$

Finally, since we have already tested `eul2mat_321`, we can perform a large number of tests for `mat2eul_321` to ensure it returns the original Euler angles.

1. Define many values for $\psi \in [-\pi, \pi]$, $\theta \in (-\pi/2, \pi/2)$, and $\phi \in [-\pi, \pi]$. Make sure not to specify $\theta = \pm\pi/2$; due to the 3-2-1 Euler angle singularity at these points, we will not be able to recover the original quaternion from the 3-2-1 Euler angles.
2. Randomly reorder the values of ψ, θ, and ϕ to ensure we have coverage for all combinations of signs.
3. For each of the values from step 1, calculate the rotation matrix using `eul2mat_321`. Then, recalculate the 3-2-1 Euler angles using `mat2eul_321` and make sure they match the original 3-2-1 Euler angles.

quat2mat (Algorithm 39)

> The test cases below are adapted from the examples in [71].

$$\mathbf{q} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

$$\mathbf{q} = \begin{bmatrix} 1 \\ 0.5 \\ 0.3 \\ 0.1 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{R} = \begin{bmatrix} 0.8519 & 0.3704 & -0.3704 \\ 0.0741 & 0.6148 & 0.7852 \\ 0.5185 & -0.6963 & 0.4963 \end{bmatrix}$$

`eul2quat_321` (Algorithm 42)

A simple numerical test is

$$\psi = \frac{\pi}{6}, \quad \theta = -\frac{\pi}{6}, \quad \phi = \frac{3\pi}{4} \qquad \Longrightarrow \mathbf{q} = \begin{bmatrix} 0.2952 \\ 0.8876 \\ 0.1353 \\ 0.3266 \end{bmatrix}$$

Additionally, since we have already tested `eul2mat_321` and `quat2mat`, we can perform a large number of tests for `eul2quat_321` using the following basic procedure:

1. Define many values for $\psi \in [-\pi, \pi]$, $\theta \in [-\pi/2, \pi/2]$, and $\phi \in [-\pi, \pi]$.
2. Randomly reorder the values of ψ, θ, and ϕ to ensure we have coverage for all combinations of signs.
3. For each of the values from step 1, calculate the rotation matrix using `eul2mat_321`; this represents the "true" rotation matrix.
4. For each of the values from step 1, first calculate the quaternion using `eul2quat_321`, and then calculate the rotation matrix using `quat2mat`; these are the rotation matrices we test against the "true" rotation matrices from step 3.

`mat2quat` (Algorithm 40)

To test `mat2quat`, we can first perform the `quat2mat` tests in reverse, but noting that the outputs of `mat2quat` will be unit quaternions.

$$\mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q} = \begin{bmatrix} \sqrt{2}/2 \\ 0 \\ \sqrt{2}/2 \\ 0 \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} 0.8519 & 0.3704 & -0.3704 \\ 0.0741 & 0.6148 & 0.7852 \\ 0.5185 & -0.6963 & 0.4963 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q} = \begin{bmatrix} 0.8607 \\ 0.4303 \\ 0.2582 \\ 0.0861 \end{bmatrix}$$

Additionally, since we have already tested `eul2mat_321` and `eul2quat_321`, we can perform a large number of tests for `mat2quat` using the following basic procedure:

1. Define many values for $\psi \in [-\pi, \pi]$, $\theta \in [-\pi/2, \pi/2]$, and $\phi \in [-\pi, \pi]$.
2. Randomly reorder the values of ψ, θ, and ϕ to ensure we have coverage for all combinations of signs.
3. For each of the values from step 1, calculate the quaternion using `eul2quat_321`; this represents the "true" quaternion.
4. For each of the values from step 1, first calculate the rotation matrix using `eul2mat_321`, and then calculate the quaternion using `quat2mat`; these are the quaternions we test against the "true" quaternions from step 3.

`quat2eul_321` (Algorithm 41)

A simple numerical test is

$$\mathbf{q} = \begin{bmatrix} 0.2952 \\ 0.8876 \\ 0.1353 \\ 0.3266 \end{bmatrix} \qquad \Longrightarrow \qquad \psi = \frac{\pi}{6}, \quad \theta = -\frac{\pi}{6}, \quad \phi = \frac{3\pi}{4}$$

Additionally, since we have already tested `eul2quat_321`, we can perform a large number of tests for `quat2eul_321` to ensure it returns the original Euler angles.

1. Define many values for $\psi \in [-\pi, \pi]$, $\theta \in (-\pi/2, \pi/2)$, and $\phi \in [-\pi, \pi]$. Make sure not to specify $\theta = \pm\pi/2$; due to the 3-2-1 Euler angle singularity at these points, we will not be able to recover the original quaternion from the 3-2-1 Euler angles.
2. Randomly reorder the values of ψ, θ, and ϕ to ensure we have coverage for all combinations of signs.
3. For each of the values from step 1, calculate the quaternion using `eul2quat_321`. Then, recalculate the 3-2-1 Euler angles using `quat2eul_321` and make sure they match the original 3-2-1 Euler angles.

`axang2quat` (Algorithm **43**)

Next, we unit test `axang2quat` since it is an extremely simple algorithm. We can use the following numerical test cases:

$$\mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = 0 \quad \implies \quad \mathbf{q} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = \pi \quad \implies \quad \mathbf{q} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{e} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}, \quad \Phi = 0 \quad \implies \quad \mathbf{q} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{e} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}, \quad \Phi = \pi \quad \implies \quad \mathbf{q} = \begin{bmatrix} 0 \\ -\sqrt{3}/3 \\ -\sqrt{3}/3 \\ -\sqrt{3}/3 \end{bmatrix}$$

$$\mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = \frac{\pi}{2} \quad \implies \quad \mathbf{q} = \begin{bmatrix} \sqrt{2}/2 \\ \sqrt{2}/2 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{e} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}, \quad \Phi = \frac{\pi}{2} \quad \implies \quad \mathbf{q} = \begin{bmatrix} \sqrt{2}/2 \\ -\sqrt{6}/6 \\ -\sqrt{6}/6 \\ -\sqrt{6}/6 \end{bmatrix}$$

$$\mathbf{e} = \begin{bmatrix} 0.1 \\ 0.5 \\ -0.3 \end{bmatrix}, \quad \Phi = \frac{7\pi}{4} \quad \implies \quad \mathbf{q} = \begin{bmatrix} 0.9239 \\ -0.0647 \\ -0.3234 \\ 0.1941 \end{bmatrix}$$

`quat2axang` (Algorithm **44**)

To test `quat2axang`, we can first perform (most of) the `axang2quat` tests in reverse. Note that the first test below

tests the singularity at $\Phi = 0$.

$$\mathbf{q} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \implies \quad \mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = 0$$

$$\mathbf{q} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \implies \quad \mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = \pi$$

$$\mathbf{q} = \begin{bmatrix} 0 \\ -\sqrt{3}/3 \\ -\sqrt{3}/3 \\ -\sqrt{3}/3 \end{bmatrix}, \quad \implies \quad \mathbf{e} = \begin{bmatrix} -\sqrt{3}/3 \\ -\sqrt{3}/3 \\ -\sqrt{3}/3 \end{bmatrix}, \quad \Phi = \pi$$

$$\mathbf{q} = \begin{bmatrix} \sqrt{2}/2 \\ \sqrt{2}/2 \\ 0 \\ 0 \end{bmatrix}, \quad \implies \quad \mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = \frac{\pi}{2}$$

$$\mathbf{q} = \begin{bmatrix} \sqrt{2}/2 \\ -\sqrt{6}/6 \\ -\sqrt{6}/6 \\ -\sqrt{6}/6 \end{bmatrix}, \quad \implies \quad \mathbf{e} = \begin{bmatrix} -\sqrt{3}/3 \\ -\sqrt{3}/3 \\ -\sqrt{3}/3 \end{bmatrix}, \quad \Phi = \frac{\pi}{2}$$

$$\mathbf{q} = \begin{bmatrix} 0.3827 \\ 0.1562 \\ 0.7808 \\ -0.4685 \end{bmatrix}, \quad \implies \quad \mathbf{e} = \begin{bmatrix} 0.1690 \\ 0.8452 \\ -0.5071 \end{bmatrix}, \quad \Phi = \frac{3\pi}{4}$$

Finally, since we have already tested `axang2quat`, we can perform a large number of tests for `quat2axang`.

1. Define many values for $\Phi \in (0, 2\pi)$, $e_1 \in [-1, 1]$, $e_2 \in [-1, 1]$, and $e_3 \in [-1, 1]$. Make sure not to specify $\Phi = 0$ since this represents an edge case that is tested separately.
2. Normalize every principal rotation vector.
3. Randomly reorder the values of Φ, e_1, e_2, and e_3 to ensure we have coverage for all combinations of signs.
4. For each of the values from step 1, calculate the quaternion using `axang2quat`, and then recover the axis-angle representation using `quat2axang`. Make sure that the recovered axis-angle representations match the original axis-angle representations that we started with.

axang2mat (Algorithm 29)

To unit test `axang2mat`, we can use the following numerical test cases:

$$
\mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = 0 \quad \implies \quad \mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

$$
\mathbf{e} = \begin{bmatrix} -5 \\ 4 \\ -2 \end{bmatrix}, \quad \Phi = 0 \quad \implies \quad \mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

$$
\mathbf{e} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \Phi = \frac{\pi}{2} \quad \implies \quad \mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}
$$

$$
\mathbf{e} = \begin{bmatrix} 0.1 \\ 0.2 \\ -0.4 \end{bmatrix}, \quad \Phi = \frac{5\pi}{4} \quad \implies \quad \mathbf{R} = \begin{bmatrix} -0.6258 & 0.7798 & -0.0166 \\ -0.4546 & -0.3819 & -0.8046 \\ -0.6338 & -0.4960 & 0.5935 \end{bmatrix}
$$

Additionally, since we have already tested `axang2quat` and `quat2mat`, we can perform a large number of tests for `axang2mat`.

1. Define many values for $\Phi \in [0, 2\pi]$, $e_1 \in [-1, 1]$, $e_2 \in [-1, 1]$, and $e_3 \in [-1, 1]$.
2. Randomly reorder the values of Φ, e_1, e_2, and e_3 to ensure we have coverage for all combinations of signs.
3. For each of the values from step 1, calculate the quaternion using `axang2quat`. Then, calculate the corresponding rotation matrix using `quat2mat`; this represents the "true" rotation matrix.
4. For each of the values from step 1, calculate the rotation matrix using `axang2mat`; these are the rotation matrices we test against the "true" rotation matrices from step 3.

`mat2axang` (Algorithm 28)

To test `mat2axang`, we can first perform the third and fourth `axang2mat` tests in reverse. Note that we skip the first two tests since the rotation matrix to axis-angle conversion has a singularity at $\Phi = 0$ that we will test separately.

$$
\mathbf{R} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad \implies \quad \mathbf{e} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \Phi = \frac{\pi}{2}
$$

$$
\mathbf{R} = \begin{bmatrix} -0.6258 & -0.4546 & -0.6338 \\ 0.7798 & -0.3819 & -0.4960 \\ -0.0166 & -0.8046 & 0.5935 \end{bmatrix} \quad \implies \quad \mathbf{e} = \begin{bmatrix} 0.1 \\ 0.2 \\ -0.4 \end{bmatrix}, \quad \Phi = \frac{3\pi}{4}
$$

Next, we test the $\Phi = 0$ singularity. Note that the default behavior for this case is to return $(1, 0, 0)^T$ for the principal rotation vector, since there is no way to determine the principal rotation vector if a rotation does not occur. Additionally, since `axang2mat` has already been tested at this point, we can use it to when defining this test case.

$$\mathbf{e} = (0.2673, 0.5345, 0.8018)^T$$
$$\Phi = 0$$
$$\mathbf{R} = \texttt{axang2mat}(\mathbf{e}, \Phi)$$
$$\texttt{mat2axang}(\mathbf{R}) \quad \implies \quad \mathbf{e} = (1, 0, 0)^T, \quad \Phi = 0$$

Next, we test the case where $\Phi = \pi$, which is not a singularity, but does represent an edge case for `mat2axang`.

$$\mathbf{e} = (0.2673, 0.5345, 0.8018)^T$$
$$\Phi = \pi$$
$$\mathbf{R} = \texttt{axang2mat}(\mathbf{e}, \Phi)$$
$$\texttt{mat2axang}(\mathbf{R}) \quad \implies \quad \mathbf{e} = (0.2673, 0.5345, 0.8018)^T, \quad \Phi = \pi$$

Finally, since we have already tested `axang2mat`, we can perform a large number of tests for `mat2axang`.

1. Define many values for $\Phi \in (0, 2\pi)$, $e_1 \in [-1, 1]$, $e_2 \in [-1, 1]$, and $e_3 \in [-1, 1]$. Make sure not to specify $\Phi = 0$ or $\Phi = 2\pi$ since these represent edge cases that are test separately.
2. Normalize every principal rotation vector.
3. Randomly reorder the values of Φ, e_1, e_2, and e_3 to ensure we have coverage for all combinations of signs.
4. For each of the values from step 1, calculate the rotation matrix using `axang2mat`, and then recover the axis-angle representation using `mat2axang`. Make sure that the recovered axis-angle representations match the original axis-angle representations that we started with.

`axang2eul_321` (Algorithm 30)

To unit test `axang2mat`, we can use the following numerical test cases:

$$\mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = 0 \quad \implies \quad \psi = 0, \quad \theta = 0, \quad \phi = 0$$

$$\mathbf{e} = \begin{bmatrix} -5 \\ 4 \\ -2 \end{bmatrix}, \quad \Phi = 0 \quad \implies \quad \psi = 0, \quad \theta = 0, \quad \phi = 0$$

$$\mathbf{e} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \Phi = \frac{\pi}{2} \quad \implies \quad \psi = 0, \quad \theta = \frac{\pi}{2}, \quad \phi = 0$$

$$\mathbf{e} = \begin{bmatrix} 0.1 \\ 0.2 \\ -0.4 \end{bmatrix}, \quad \Phi = \frac{3\pi}{4} \quad \implies \quad \psi = 2.2471, \quad \theta = 0.0166, \quad \phi = -0.9352$$

Additionally, since we have already tested `axang2mat` and `mat2eul_321`, we can perform a large number of tests for `axang2eul_321`.

1. Define many values for $\Phi \in [0, 2\pi]$, $e_1 \in [-1, 1]$, $e_2 \in [-1, 1]$, and $e_3 \in [-1, 1]$.
2. Randomly reorder the values of Φ, e_1, e_2, and e_3 to ensure we have coverage for all combinations of signs.
3. For each of the values from step 1, calculate the rotation matrix using `axang2mat`. Then, calculate the corresponding 3-2-1 Euler angles using `quat2eul_321`; these represent the "true" 3-2-1 Euler angles.
4. For each of the values from step 1, calculate the 3-2-1 Euler angles using `axang2eul_321`; these are the 3-2-1 Euler angles we test against the "true" 3-2-1 Euler angles from step 3.

`eul2axang_321` (Algorithm 31)

The final rotation parameterization conversion algorithm we need to test is `eul2axang_321`. Recall from Section 4.4.2 that we first use `eul2quat_321` to obtain a quaternion, and then use `quat2axang` to convert that quaternion to the axis-angle representation. Therefore, instead of doing a large suite of a tests, we instead just do a few simple numerical tests to make sure we wrapped around these algorithms correctly.

$$\psi = 0, \quad \theta = 0, \quad \phi = 0 \quad \implies \quad \mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Phi = 0$$

$$\psi = \frac{\pi}{4}, \quad \theta = \frac{\pi}{8}, \quad \phi = -\frac{\pi}{6} \quad \implies \quad \mathbf{e} = \begin{bmatrix} -0.5930 \\ 0.1488 \\ 0.7913 \end{bmatrix}, \quad \Phi = 1.0869$$

A.1.3 Quaternions

quatmul (Algorithm 33)

First, let's define the quaternions $\mathbf{p}$, $\mathbf{q}$, and $\mathbf{r}$.

$$\mathbf{p} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \qquad \mathbf{q} = \begin{bmatrix} 1 \\ 0.5 \\ 0.5 \\ 0.75 \end{bmatrix}, \qquad \mathbf{r} = \begin{bmatrix} 2 \\ 1 \\ 0.1 \\ 0.1 \end{bmatrix}$$

We can then define the following simple numerical test cases [74]:

$$\mathbf{p} \otimes \mathbf{p} = \texttt{quatmul}(\mathbf{p}, \mathbf{p}) = \begin{bmatrix} 0 \\ 0 \\ 2 \\ 0 \end{bmatrix}$$

$$\mathbf{p} \otimes \mathbf{q} = \texttt{quatmul}(\mathbf{p}, \mathbf{q}) = \begin{bmatrix} 0.5 \\ 1.25 \\ 1.5 \\ 0.25 \end{bmatrix}$$

$$\mathbf{p} \otimes \mathbf{r} = \texttt{quatmul}(\mathbf{p}, \mathbf{r}) = \begin{bmatrix} 1.9 \\ 1.1 \\ 2.1 \\ -0.9 \end{bmatrix}$$

quatconj (Algorithm 32)

$$\mathbf{q} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q}^* = \texttt{quatconj}(\mathbf{q}) = \begin{bmatrix} 1 \\ -2 \\ -3 \\ -4 \end{bmatrix}$$

$$\mathbf{q} = \begin{bmatrix} 1 \\ -2 \\ -3 \\ -4 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q}^* = \texttt{quatconj}(\mathbf{q}) = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\mathbf{q} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q}^* = \texttt{quatconj}(\mathbf{q}) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

quatnorm (Algorithm 34)

$$\mathbf{q} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \implies \|\mathbf{q}\| = \texttt{quatnorm}(\mathbf{q}) = 5.4772$$

$$\mathbf{q} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \implies \|\mathbf{q}\| = \texttt{quatnorm}(\mathbf{q}) = 2$$

$$\mathbf{q} = \begin{bmatrix} 0 \\ 1 \\ -1 \\ -1 \end{bmatrix} \implies \|\mathbf{q}\| = \texttt{quatnorm}(\mathbf{q}) = \sqrt{3}$$

$$\mathbf{q} = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0 \end{bmatrix} \implies \|\mathbf{q}\| = \texttt{quatnorm}(\mathbf{q}) = 1$$

`quatnormalize` (Algorithm 36)

First, we can perform a set of simple numerical tests.

$$\mathbf{q} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \implies \frac{\mathbf{q}}{\|\mathbf{q}\|} = \texttt{quatnormalize}(\mathbf{q}) = \begin{bmatrix} 0.1826 \\ 0.3651 \\ 0.5477 \\ 0.7303 \end{bmatrix}$$

$$\mathbf{q} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \implies \frac{\mathbf{q}}{\|\mathbf{q}\|} = \texttt{quatnormalize}(\mathbf{q}) = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}$$

$$\mathbf{q} = \begin{bmatrix} 0 \\ 1 \\ -1 \\ -1 \end{bmatrix} \implies \frac{\mathbf{q}}{\|\mathbf{q}\|} = \texttt{quatnormalize}(\mathbf{q}) = \begin{bmatrix} 0 \\ \sqrt{3}/3 \\ -\sqrt{3}/3 \\ -\sqrt{3}/3 \end{bmatrix}$$

$$\mathbf{q} = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0 \end{bmatrix} \implies \frac{\mathbf{q}}{\|\mathbf{q}\|} = \texttt{quatnormalize}(\mathbf{q}) = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0 \end{bmatrix}$$

Additionally, since we have already tested `quatnorm` and `eul2quat_321`, we can perform a large number of tests for `quatnormalize` using the following basic procedure:

1. Define many values for $q_0, q_1, q_2, q_3 \in [-10, 10]$.
2. Randomly reorder the values of q_0, q_1, q_2, and q_3 to ensure we have coverage for all kinds of rotations.
3. Normalize each randomly generated quaternion.
4. Take the norm of each of the randomly generated quaternions using `quatnorm`.
5. Check to make sure all normalized quaternions actually have a norm of 1.

`quatinv` (Algorithm 35)

First, we can perform a set of simple numerical tests.

$$
\mathbf{q} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q}^{-1} = \texttt{quatinv}(\mathbf{q}) = \begin{bmatrix} 1/30 \\ -1/15 \\ -1/10 \\ -26/195 \end{bmatrix}
$$

$$
\mathbf{q} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q}^{-1} = \texttt{quatinv}(\mathbf{q}) = \begin{bmatrix} 0.25 \\ -0.25 \\ -0.25 \\ -0.25 \end{bmatrix}
$$

$$
\mathbf{q} = \begin{bmatrix} 0 \\ 1 \\ -1 \\ -1 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q}^{-1} = \texttt{quatinv}(\mathbf{q}) = \begin{bmatrix} 0 \\ -1/3 \\ 1/3 \\ 1/3 \end{bmatrix}
$$

$$
\mathbf{q} = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q}^{-1} = \texttt{quatinv}(\mathbf{q}) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}
$$

Additionally, since we have already tested `quatmul`, we can perform a large number of tests for `quatinv` using the following basic procedure:

1. Define many values for $q_0, q_1, q_2, q_3 \in [-10, 10]$.
2. Randomly reorder the values of q_0, q_1, q_2, and q_3 to ensure we have coverage for all kinds of rotations.
3. Find the inverse of each randomly generated quaternion using `quatinv`.
4. Multiply each randomly generated quaternion by its inverse and check to make sure it equals $(1, 0, 0, 0)^T$ (the identity quaternion).

quatchain (Algorithm 38)

First, let's just perform a simple numerical test.

$$
\mathbf{q}_{A \to B} = \begin{bmatrix} 0.1826 \\ 0.3651 \\ 0.5477 \\ 0.7303 \end{bmatrix}, \quad \mathbf{q}_{B \to C} = \begin{bmatrix} 0.2662 \\ -0.0690 \\ -0.3451 \\ 0.8973 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{q}_{A \to C} = \begin{bmatrix} 0.3925 \\ -0.8281 \\ 0.2952 \\ -0.2701 \end{bmatrix}
$$

Additionally, since we have already tested `eul2quat_321`, `eul2mat_321`, and `mat2quat`, we can perform a large number of tests for `quatchain` using the following basic procedure:

1. Define many values for $\psi_{A \to B} \in [-\pi, \pi]$, $\psi_{B \to C} \in [-\pi, \pi]$, $\theta_{A \to B} \in [-\pi/2, \pi/2]$, $\theta_{B \to C} \in [-\pi/2, \pi/2]$, $\phi_{A \to B} \in [-\pi, \pi]$, and $\phi_{B \to C} \in [-\pi, \pi]$.
2. Randomly reorder the values of ψ, θ, and ϕ to ensure we have coverage for all combinations of signs.
3. For each pair of rotations:
   (a) $\mathbf{q}_{A \to B} = \texttt{eul2quat\_321}(\psi_{A \to B}, \theta_{A \to B}, \phi_{A \to B})$
   (b) $\mathbf{q}_{B \to C} = \texttt{eul2quat\_321}(\psi_{B \to C}, \theta_{B \to C}, \phi_{B \to C})$
   (c) $\mathbf{q}_{A \to C} = \texttt{quatchain}(\mathbf{q}_{A \to B}, \mathbf{q}_{A \to C})$
   (d) $\mathbf{R}_{A \to B} = \texttt{eul2mat\_321}(\psi_{A \to B}, \theta_{A \to B}, \phi_{A \to B})$
   (e) $\mathbf{R}_{B \to C} = \texttt{eul2mat\_321}(\psi_{B \to C}, \theta_{B \to C}, \phi_{B \to C})$
   (f) $\mathbf{R}_{A \to C} = \mathbf{R}_{B \to C} \mathbf{R}_{A \to B}$
   (g) $[\mathbf{q}_{A \to C}]_{\text{true}} = \texttt{mat2quat}(\mathbf{R}_{A \to C})$
   (h) Check to make sure that $\mathbf{q}_{A \to C}$ is equal to $[\mathbf{q}_{A \to C}]_{\text{true}}$.

`quatrotate` **(Algorithm 37)**

$$\mathbf{q}_{A \to B} = \begin{bmatrix} 0.7018 \\ -0.5417 \\ 0.1724 \\ 0.4292 \end{bmatrix}, \quad [\boldsymbol{r}]_A = \begin{bmatrix} 5 \\ 4 \\ 3 \end{bmatrix} \quad \Longrightarrow \quad [\boldsymbol{r}]_B = \texttt{quatrotate}(\mathbf{q}_{A \to B}, [\boldsymbol{r}]_A) = \begin{bmatrix} 2.4016 \\ -5.6053 \\ 3.5794 \end{bmatrix}$$

`quatang` **(Algorithm 45)**

First, let's just perform a simple numerical test.

$$\mathbf{q}_1 = \begin{bmatrix} 0.9173 \\ -0.3023 \\ -0.0655 \\ 0.2508 \end{bmatrix}, \quad \mathbf{q}_{B \to C} = \begin{bmatrix} 0.5972 \\ 0.5180 \\ -0.2343 \\ 0.5658 \end{bmatrix} \quad \Longrightarrow \quad \theta = \texttt{quatang}(\mathbf{q}_1, \mathbf{q}_2) = 1.9806$$

Additionally, since we have already tested `quatmul`, `quatconj`, and `quat2axang`, we can perform a large number of tests for `quatang` using the following basic procedure, adapted from the discussion in Section 4.5.7:

1. Define many values for $q_{1,0}, q_{1,1}, q_{1,2}, q_{1,3} \in [-10, 10]$, where $\mathbf{q}_1 = (q_{1,0}, q_{1,1}, q_{1,2}, q_{1,3})^T$.
2. Randomly reorder the values of $q_{1,0}, q_{1,1}, q_{1,2}$, and $q_{1,3}$ to ensure we have coverage for all kinds of rotations.
3. Define many values for $q_{2,0}, q_{2,1}, q_{2,2}, q_{2,3} \in [-10, 10]$, where $\mathbf{q}_2 = (q_{2,0}, q_{2,1}, q_{2,2}, q_{2,3})^T$.
4. Randomly reorder the values of $q_{2,0}, q_{2,1}, q_{2,2}$, and $q_{2,3}$ to ensure we have coverage for all kinds of rotations.
5. Normalize each quaternion.
6. For each pair of unit quaternions $(\mathbf{q}_1, \mathbf{q}_2)$:
   (a) $\mathbf{q} = \texttt{quatmul}(\texttt{quatconj}(\mathbf{q}_1), \mathbf{q}_2)$
   (b) $[\sim, \Phi] = \texttt{quat2axang}(\mathbf{q})$
   (c) $\theta = \texttt{quatang}(\mathbf{q}_1, \mathbf{q}_2)$
   (d) Check to make sure that $\theta = \Phi$.

`quatslerp` **(Algorithm 46)**

First, we define a set of three tests for the case where $\mathbf{q}_1^T \mathbf{q}_2 \geq 0$. In this case, the base SLERP equations with no modifications would automatically perform interpolation on the shortest arc.

$$\mathbf{q}_1 = \begin{bmatrix} 0.9173 \\ -0.3023 \\ -0.0655 \\ 0.2508 \end{bmatrix}, \quad \mathbf{q}_2 = \begin{bmatrix} 0.5972 \\ 0.5180 \\ -0.2343 \\ 0.5658 \end{bmatrix}$$

$$\texttt{quatslerp}(\mathbf{q}_1, \mathbf{q}_2, 0) = \mathbf{q}_1$$

$$\texttt{quatslerp}(\mathbf{q}_1, \mathbf{q}_2, 1) = \mathbf{q}_2$$

$$\texttt{quatslerp}(\mathbf{q}_1, \mathbf{q}_2, 0.2) = \begin{bmatrix} 0.9215 \\ -0.1355 \\ -0.1109 \\ 0.3467 \end{bmatrix}$$

Next, we define three more tests, this time stressing the case where $\mathbf{q}_1^T \mathbf{q}_2 < 0$, in which case the base SLERP equations

would perform interpolation on the longer path if we did not include additional logic in Algorithm 46.

$$\mathbf{q}_1 = \begin{bmatrix} 0.9173 \\ 0.3023 \\ 0.0655 \\ 0.2508 \end{bmatrix}, \qquad \mathbf{q}_2 = \begin{bmatrix} 0.1826 \\ -0.3651 \\ -0.5477 \\ -0.7303 \end{bmatrix}$$

$$\texttt{quatslerp}(\mathbf{q}_1, \mathbf{q}_2, 0) = \mathbf{q}_1$$

$$\texttt{quatslerp}(\mathbf{q}_1, \mathbf{q}_2, 1) = \mathbf{q}_2$$

$$\texttt{quatslerp}(\mathbf{q}_1, \mathbf{q}_2, 0.2) = \begin{bmatrix} 0.0913 \\ 0.4192 \\ 0.5196 \\ 0.7389 \end{bmatrix}$$

A.2   Time Test Cases

A.2.1   Time Units

cal2doy (Algorithm 65) ↔ **doy2cal (Algorithm 66)**

Date	Day of Year	Description
2022 January 22	22	date in January
2020 March 18	78	date in leap year after January
2020 December 31	366	last day of year in leap year
2022 January 1	1	first day of year
2022 December 31	365	last day of year in non-leap year

These test cases were verified using `https://www.esrl.noaa.gov/gmd/grad/neubrew/Calendar.jsp`.

cal2mjd (Algorithm 71) ↔ **mjd2cal (Algorithm 72)**

Gregorian Date	Modified Julian Date
1582 October 15 00:00:00	-100840
1600 January 1 00:00:00	-94553
1600 January 1 06:00:00	-94552.75
1600 January 1 12:00:00	-94552.5
1600 January 1 18:00:00	-94552.25
1858 November 16 18:00:00	-0.25
1858 November 17 00:00:00	0
1858 November 17 06:00:00	0.25
2000 January 1 12:00:00	51544.5
2005 May 24 00:00:00	53514
2006 December 19 00:00:00	54088

2006 December 19 06:00:00	54088.25
2006 December 19 18:00:00	54088.75

> These test cases were verified using https://planetcalc.com/503/ and https://heasarc.gsfc.nasa.gov/cgi-bin/Tools/xTime/xTime.pl. Note that some of these test cases also came directly from [57].

Additionally, based on the definitions of `cal2mjd` and `mjd2cal` in Algorithms 71 and 72, respectively, their implementations should be tested to make sure they raise an error if a date before 15 October 1582 (-100840 MJD) is given.

`f2hms` (Algorithm **74**)

$(0.524223$ of a day$) = 12{:}34{:}52.867199$ (to 6 decimal places)

`hms2f` (Algorithm **73**)

$12 : 34 : 52.890204 = (0.524223$ of a day$)$ (to 6 decimal places)

`jd2mjd` (Algorithm **67**) $\leftrightarrow$ `mjd2jd` (Algorithm **68**)

JD	MJD
0	-2400000.5
100	-2399900.5
2400000.5	0
2400100.5	100

`jd2t` (Algorithm **69**)

1992 August 20 12:14:00 (JD 2448855.009722222) is equal to -0.073647919 Julian centuries since J2000.0.

> This test case was adapted from Example 3-5 in [103, p. 188].

`mjd2f` (Algorithm **75**)

Date	Day of Year	Description
1858 November 11 15:50:24	0.66	negative MJD
1858 November 16 15:50:24	0.66	barely negative MJD
1858 November 17 16:04:48	0.67	barely positive MJD
2018 July 22 16:04:48	0.67	positive MJD

`mjd2t` (Algorithm **70**)

1992 August 20 12:14:00 (MJD 48854.50972222222) is equal to -0.073647919 Julian centuries since J2000.0.

> This test case was adapted from Example 3-5 in [103, p. 188].

A.2.2 Time Scales

`get_dat` (Algorithm 87)

Date	Modified Julian Date	ΔAT [s]	Description
1972 January 1	41317	10	inside data range
1972 June 30	41498	10	inside data range
1972 July 1	41499	11	inside data range
2005 January 1	53371	32	inside data range
2005 January 2	53372	32	inside data range
2005 December 31	53735	32	inside data range
2006 January 1	53736	33	inside data range
1971 December 31	41316	10	outside data range
1900 January 1	15020	10	outside data range
2017 January 2	57755	37	outside data range
2023 May 7	60071	37	outside data range

`get_dut1` (Algorithm 86)

Date	Modified Julian Date	$\Delta UT1$ [s]	Description
1992 January 1	48622	-0.1251659	inside data range
2004 July 25	53211	-0.4573568	inside data range
2017 December 23	58110	0.2252297	inside data range
1991 December 31	48621	-0.1251659	outside data range

> These test cases were randomly selected from `https://datacenter.iers.org/data/html/finals2000A.data.html`.

A.3 Angle Test Cases

`arcsec2deg` (Algorithm 91) $\leftrightarrow$ `deg2arcsec` (Algorithm 90)

$\theta°$	θ''

5.4321°	19555.56
6.515555555555555°	23456

arcsec2rad (Algorithm 93)   ↔   **rad2arcsec (Algorithm 92)**

θ^{rad}	θ''
0.1 rad	20626.48062470964
0.096962736221907 rad	20000

deg2dms (Algorithm 95)   ↔   **dms2deg (Algorithm 94)**

$-35° - 15' - 53.63'' = -35.264897°$

> This test case was adapted from Example 3-8 in [103, p. 198].

dms2rad (Algorithm 96)   ↔   **rad2dms (Algorithm 97)**

$-35° - 15' - 53.63'' = -0.6154886$ rad

> This test case was adapted from Example 3-8 in [103, p. 198].

deg2rad (Algorithm 88)   ↔   **rad2deg (Algorithm 89)**

θ^{rad}	$\theta°$
$-\pi/4$ rad	$-45°$
0 rad	0°
$\pi/4$ rad	45°
2π rad	360°
4π rad	720°

A.4   Kinematics Test Cases

add_ang_acc (Algorithm 12)

$\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}}$	$\left[^{\mathcal{B}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{B}}$	$\left[^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right]_{\mathcal{C}}$	$\left[^{\mathcal{C}/\mathcal{B}}\boldsymbol{\alpha}\right]_{\mathcal{C}}$	$\mathbf{R}_{\mathcal{B}\to\mathcal{C}}$	$\left[^{\mathcal{C}/\mathcal{A}}\boldsymbol{\alpha}\right]_{\mathcal{C}}$

$\begin{bmatrix}0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\end{bmatrix}$	$\begin{bmatrix}1\\2\\3\end{bmatrix}$	$\mathbf{I}_{3\times3}$	$\begin{bmatrix}1\\2\\3\end{bmatrix}$
$\begin{bmatrix}1\\2\\3\end{bmatrix}$	$\begin{bmatrix}4\\5\\6\end{bmatrix}$	$\begin{bmatrix}7\\8\\9\end{bmatrix}$	$\begin{bmatrix}1\\2\\3\end{bmatrix}$	$\begin{bmatrix}0.3536 & 0.6124 & -0.7071\\ -0.4928 & 0.7645 & 0.4156\\ 0.7951 & 0.2015 & 0.5721\end{bmatrix}$	$\begin{bmatrix}-1.5344\\31.6318\\-9.7037\end{bmatrix}$

`add_ang_vel` (Algorithm 11)

$\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}}$	$\left[{}^{\mathcal{C}/\mathcal{B}}\boldsymbol{\omega}\right]_{\mathcal{C}}$	$\mathbf{R}_{\mathcal{B}\rightarrow\mathcal{C}}$	$\left[{}^{\mathcal{C}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{C}}$
$\begin{bmatrix}0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\end{bmatrix}$	$\mathbf{I}_{3\times3}$	$\begin{bmatrix}0\\0\\0\end{bmatrix}$
$\begin{bmatrix}1\\2\\3\end{bmatrix}$	$\begin{bmatrix}4\\5\\6\end{bmatrix}$	$\mathbf{I}_{3\times3}$	$\begin{bmatrix}5\\7\\9\end{bmatrix}$
$\begin{bmatrix}1\\2\\3\end{bmatrix}$	$\begin{bmatrix}4\\5\\6\end{bmatrix}$	$\begin{bmatrix}0.3536 & 0.6124 & -0.7071\\ -0.4928 & 0.7645 & 0.4156\\ 0.7951 & 0.2015 & 0.5721\end{bmatrix}$	$\begin{bmatrix}3.4571\\7.2830\\8.9144\end{bmatrix}$

`matderiv` (Algorithm 10)

$\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}$	$\left[{}^{\mathcal{B}/\mathcal{A}}\boldsymbol{\omega}\right]_{\mathcal{B}}$	$\dfrac{d\mathbf{R}_{\mathcal{A}\rightarrow\mathcal{B}}}{dt}$
$\mathbf{I}_{3\times3}$	$\begin{bmatrix}0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}$
$\mathbf{I}_{3\times3}$	$\begin{bmatrix}1\\2\\3\end{bmatrix}$	$\begin{bmatrix}0 & 3 & -2\\ -3 & 0 & 1\\ 2 & -1 & 0\end{bmatrix}$
$\begin{bmatrix}0.3536 & 0.6124 & -0.7071\\ -0.4928 & 0.7645 & 0.4156\\ 0.7951 & 0.2015 & 0.5721\end{bmatrix}$	$\begin{bmatrix}1\\2\\3\end{bmatrix}$	$\begin{bmatrix}-3.0686 & 1.8905 & 0.1026\\ -0.2657 & -1.6357 & 2.6934\\ 1.2000 & 0.4603 & -1.8298\end{bmatrix}$

`rv2omega` (Algorithm 25)

$$[\boldsymbol{r}]_{\mathcal{I}} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \text{ m}$$

$$[\boldsymbol{v}]_{\mathcal{I}} = \begin{bmatrix} -3 \\ -2 \\ -1 \end{bmatrix} \text{ m/s}$$

$$\texttt{rv2omega}\,([\boldsymbol{r}]_{\mathcal{I}}, [\boldsymbol{v}]_{\mathcal{I}}) \quad \Longrightarrow \quad \left[^{\mathcal{R}/\mathcal{I}}\boldsymbol{\omega}\right]_{\mathcal{I}} = \begin{bmatrix} 2/7 \\ -4/7 \\ 2/7 \end{bmatrix} \text{ rad/s}$$

`skew2vec` **(Algorithm 7)**

$$\mathbf{a} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \quad \Longrightarrow \quad \mathbf{a}^{\times} = \begin{bmatrix} 0 & -3 & 2 \\ 3 & 0 & -1 \\ -2 & 1 & 0 \end{bmatrix} \quad \Longrightarrow \quad \mathbf{a} = \texttt{skew2vec}(\mathbf{a}^{\times}) = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

`vec2skew` **(Algorithm 6)**

$$\mathbf{a} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \quad \Longrightarrow \quad \mathbf{a}^{\times} = \texttt{vec2skew}(\mathbf{a}) = \begin{bmatrix} 0 & -3 & 2 \\ 3 & 0 & -1 \\ -2 & 1 & 0 \end{bmatrix}$$

A.5 Orbital Mechanics Test Cases

A.5.1 Orbits in Two Dimensions

`a2T` **(Algorithm 55)**

$$a = 7 \times 10^6 \text{ m}$$
$$\mu = 3.986004418 \times 10^{14} \text{ m}^3/\text{s}^2$$
$$\texttt{a2T}(a, \mu) \quad \Longrightarrow \quad a = 5.828516637686015 \times 10^3 \text{ m}$$

`rot_pqw2rsw` **and** **`rot_rsw2pqw`** **(Algorithms 52 and 53)**

θ [rad]	$\mathbf{R}_{\text{pqw}\to\text{rsw}}$ *rot_pqw2rsw*	$\mathbf{R}_{\text{rsw}\to\text{pqw}}$ *rot_rsw2pqw*

	(Algorithm 52)	*(Algorithm 53)*
0	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\pi/6$	$\begin{bmatrix} \sqrt{3}/2 & 1/2 & 0 \\ -1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} \sqrt{3}/2 & -1/2 & 0 \\ 1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\pi/4$	$\begin{bmatrix} \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ -\sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\pi/3$	$\begin{bmatrix} 1/2 & \sqrt{3}/2 & 0 \\ -\sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1/2 & -\sqrt{3}/2 & 0 \\ \sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\pi/2$	$\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$2\pi/3$	$\begin{bmatrix} 1/2 & \sqrt{3}/2 & 0 \\ -\sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1/2 & -\sqrt{3}/2 & 0 \\ \sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$3\pi/4$	$\begin{bmatrix} -\sqrt{2}/2 & \sqrt{2}/2 & 0 \\ -\sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$5\pi/6$	$\begin{bmatrix} -\sqrt{3}/2 & 1/2 & 0 \\ -1/2 & -\sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{3}/2 & -1/2 & 0 \\ 1/2 & -\sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
π	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$7\pi/6$	$\begin{bmatrix} -\sqrt{3}/2 & -1/2 & 0 \\ 1/2 & -\sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{3}/2 & 1/2 & 0 \\ -1/2 & -\sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$5\pi/4$	$\begin{bmatrix} -\sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -\sqrt{2}/2 & \sqrt{2}/2 & 0 \\ -\sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$4\pi/3$	$\begin{bmatrix} -1/2 & -\sqrt{3}/2 & 0 \\ \sqrt{3}/2 & -1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1/2 & \sqrt{3}/2 & 0 \\ -\sqrt{3}/2 & -1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$3\pi/2$	$\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$5\pi/3$	$\begin{bmatrix} 1/2 & -\sqrt{3}/2 & 0 \\ \sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1/2 & \sqrt{3}/2 & 0 \\ -\sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$
\begin{array}{c|cc}
7\pi/4 & \begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \begin{bmatrix} \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ -\sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
\\
11\pi/6 & \begin{bmatrix} \sqrt{3}/2 & -1/2 & 0 \\ 1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \begin{bmatrix} \sqrt{3}/2 & 1/2 & 0 \\ -1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
\\
2\pi & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{array}
$$

`rv2e_vec` (Algorithm 49)

$$[r]_{\mathcal{I}} = \begin{bmatrix} 5053 \\ -2276 \\ -5182 \end{bmatrix} \text{ m}$$

$$[v]_{\mathcal{I}} = \begin{bmatrix} 113286 \\ 181566 \\ 48281 \end{bmatrix} \text{ m/s}$$

$\mu = 3.986004418 \times 10^{14} \text{ m}^3/\text{s}^2$

$$\texttt{rv2e\_vec}\left([r]_{\mathcal{I}}, [v]_{\mathcal{I}}, \mu\right) \quad \Longrightarrow \quad [e]_{\mathcal{I}} = \begin{bmatrix} -0.029973965190951 \\ 0.066603000645626 \\ 0.068285909115768 \end{bmatrix}$$

`rv2e` (Algorithm 51)

$$[r]_{\mathcal{I}} = \begin{bmatrix} 5053 \\ -2276 \\ -5182 \end{bmatrix} \text{ m}$$

$$[v]_{\mathcal{I}} = \begin{bmatrix} 113286 \\ 181566 \\ 48281 \end{bmatrix} \text{ m/s}$$

$\mu = 3.986004418 \times 10^{14} \text{ m}^3/\text{s}^2$

$$\texttt{rv2e}\left([r]_{\mathcal{I}}, [v]_{\mathcal{I}}, \mu\right) \quad \Longrightarrow \quad e = 0.099986817471288$$

`rv2fpa` (Algorithm 54)

First, we test for a positive value of ϕ_{fpa}. This first test is adapted from the example in [13, p. 18].

$$[r]_{\mathcal{I}} = \begin{bmatrix} 4.1852 \\ 6.2778 \\ 10.463 \end{bmatrix} \times 10^7 \text{ m}$$

$$[v]_{\mathcal{I}} = \begin{bmatrix} 2.5936 \\ 5.1872 \\ 0 \end{bmatrix} \times 10^4 \text{ m/s}$$

$$\texttt{rv2fpa}\left([r]_{\mathcal{I}}, [v]_{\mathcal{I}}\right) \quad \Longrightarrow \quad \phi_{\mathrm{fpa}} = 0.6192$$

Next, we test for a negative value of ϕ_{fpa}.

$$[r]_{\mathcal{I}} = \begin{bmatrix} 5053 \\ -2276 \\ -5182 \end{bmatrix} \text{ m}$$

$$[v]_{\mathcal{I}} = \begin{bmatrix} 113286 \\ 181566 \\ 48281 \end{bmatrix} \text{ m/s}$$

$$\texttt{rv2fpa}\left([r]_{\mathcal{I}}, [v]_{\mathcal{I}}\right) \quad \Longrightarrow \quad \phi_{\mathrm{fpa}} = -0.054698152803929$$

rvh2e_vec (Algorithm 48)

$$[r]_{\mathcal{I}} = \begin{bmatrix} 5053 \\ -2276 \\ -5182 \end{bmatrix} \text{ m}$$

$$[v]_{\mathcal{I}} = \begin{bmatrix} 113286 \\ 181566 \\ 48281 \end{bmatrix} \text{ m/s}$$

$$[h]_{\mathcal{I}} = \begin{bmatrix} 830987456 \\ -831011945 \\ 1175291934 \end{bmatrix} \text{ m}^2/\text{s}$$

$$\mu = 3.986004418 \times 10^{14} \text{ m}^3/\text{s}^2$$

$$\texttt{rvh2e\_vec}\left([r]_{\mathcal{I}}, [v]_{\mathcal{I}}, [h]_{\mathcal{I}}, \mu\right) \quad \Longrightarrow \quad [e]_{\mathcal{I}} = \begin{bmatrix} -0.029973965190951 \\ 0.066603000645626 \\ 0.068285909115768 \end{bmatrix}$$

rvh2e (Algorithm 50)

$$[\boldsymbol{r}]_{\mathcal{I}} = \begin{bmatrix} 5053 \\ -2276 \\ -5182 \end{bmatrix} \text{ m}$$

$$[\boldsymbol{v}]_{\mathcal{I}} = \begin{bmatrix} 113286 \\ 181566 \\ 48281 \end{bmatrix} \text{ m/s}$$

$$[\boldsymbol{h}]_{\mathcal{I}} = \begin{bmatrix} 830987456 \\ -831011945 \\ 1175291934 \end{bmatrix} \text{ m}^2/\text{s}$$

$$\mu = 3.986004418 \times 10^{14} \text{ m}^3/\text{s}^2$$

$$\texttt{rvh2e}\left([\boldsymbol{r}]_{\mathcal{I}}, [\boldsymbol{v}]_{\mathcal{I}}, [\boldsymbol{h}]_{\mathcal{I}}, \mu\right) \implies e = 0.099986817471287$$

`T2a` (Algorithm 56)

$$T = 5400 \text{ s}$$

$$\mu = 3.986004418 \times 10^{14} \text{ m}^3/\text{s}^2$$

$$\texttt{T2a}(T, \mu) \implies a = 6.652555701327529 \times 10^6 \text{ m}$$

A.6   Gravity Test Cases

`gravity_coeff_index` (Algorithm 121)

n	m	1-based indexing l	0-based indexing l
0	0	1	0
1	0	2	1
1	1	3	2
2	0	4	3
2	1	5	4
2	2	6	5
3	0	7	6
3	1	8	7
3	2	9	8
3	3	10	9
4	0	11	10
120	53	7314	7313

B

Derivations and Verifications

B.1   Derivation: 3-2-1 Rotation Sequence

derivation_3_2_1_rotation_sequence.mlx

Aerospace Simulation Toolbox

Passive rotation matrix for 3-2-1 passive rotation sequence.

- Author: Tamas Kis
- Last Update: 2023-06-19

```
% clears Workspace and Command Window, closes all figures
clear; clc; close all;
```

Rotation matrix for 3-2-1 passive rotation sequence.

Initializes the symbolic variables.

```
syms theta_1 theta_2 theta_3;
```

1st rotation about 3rd axis.

```
R3 = [ cos(theta_1)   sin(theta_1)   0;
      -sin(theta_1)   cos(theta_1)   0;
       0              0              1];
```

2nd rotation about 2nd axis.

```
R2 = [cos(theta_2)   0  -sin(theta_2);
      0              1   0;
      sin(theta_2)   0   cos(theta_2)];
```

3rd rotation about 1st axis.

```
R1 = [1   0               0;
      0   cos(theta_3)    sin(theta_3);
      0  -sin(theta_3)    cos(theta_3)];
```

3-2-1 rotation matrix.

```
R_321 = R1*R2*R3
```

```
R_321 =
```

1

$$
\begin{pmatrix}
\cos(\theta_1)\cos(\theta_2) & \cos(\theta_2)\sin(\theta_1) & -\sin(\theta_2) \\
\cos(\theta_1)\sin(\theta_2)\sin(\theta_3) - \cos(\theta_3)\sin(\theta_1) & \cos(\theta_1)\cos(\theta_3) + \sin(\theta_1)\sin(\theta_2)\sin(\theta_3) & \cos(\theta_2)\sin(\theta_3) \\
\sin(\theta_1)\sin(\theta_3) + \cos(\theta_1)\cos(\theta_3)\sin(\theta_2) & \cos(\theta_3)\sin(\theta_1)\sin(\theta_2) - \cos(\theta_1)\sin(\theta_3) & \cos(\theta_2)\cos(\theta_3)
\end{pmatrix}
$$

B.2   Derivation: 3-1-3 Rotation Sequence

derivation_3_1_3_rotation_sequence.mlx

Aerospace Simulation Toolbox

Passive rotation matrix for 3-1-3 passive rotation sequence.

- Author: Tamas Kis
- Last Update: 2023-06-19

```
% clears Workspace and Command Window, closes all figures
clear; clc; close all;
```

Rotation matrix for 3-1-3 passive rotation sequence.

Initializes the symbolic variables.

```
syms theta_1 theta_2 theta_3;
```

1st rotation about 3rd axis.

```
R3_a = [ cos(theta_1)   sin(theta_1)   0;
        -sin(theta_1)   cos(theta_1)   0;
         0              0              1];
```

2nd rotation about 1st axis.

```
R1 = [1   0              0;
      0   cos(theta_2)   sin(theta_2);
      0  -sin(theta_2)   cos(theta_2)];
```

3rd rotation about 3rd axis.

```
R3_b = [ cos(theta_3)   sin(theta_3)   0;
        -sin(theta_3)   cos(theta_3)   0;
         0              0              1];
```

3-1-3 rotation matrix.

```
R_321 = R3_b*R1*R3_a
```

```
R_321 =
```

1

$$\begin{pmatrix} \cos(\theta_1)\cos(\theta_3) - \cos(\theta_2)\sin(\theta_1)\sin(\theta_3) & \cos(\theta_3)\sin(\theta_1) + \cos(\theta_1)\cos(\theta_2)\sin(\theta_3) & \sin(\theta_2)\sin(\theta_3) \\ -\cos(\theta_1)\sin(\theta_3) - \cos(\theta_2)\cos(\theta_3)\sin(\theta_1) & \cos(\theta_1)\cos(\theta_2)\cos(\theta_3) - \sin(\theta_1)\sin(\theta_3) & \cos(\theta_3)\sin(\theta_2) \\ \sin(\theta_1)\sin(\theta_2) & -\cos(\theta_1)\sin(\theta_2) & \cos(\theta_2) \end{pmatrix}$$

B.3 Verification: Rotation Matrix Time Derivative

verification_rotation_matrix_time_derivative.mlx

Aerospace Simulation Toolbox

Example verifying the analytical expression for the time derivative of a rotation matrix in terms of the angular velocity.

- Author: Tamas Kis
- Last Update: 2023-06-18

```
% clears Workspace and Command Window, closes all figures
clear; clc; close all;
```

Initial conditions.

Angular velocity of frame B with respect to frame A, expressed in frame B.

```
w_A2B_B = [1;2;3];
```

Initial attitude of frame B with respect to frame A, represented as a quaternion.

```
q_A2B_0 = [1;2;3;4]/norm([1;2;3;4]);
```

Solving for the quaternion and rotation matrix as a function of time.

Ordinary differential equation defining the quaternion kinematics.

```
dqdt = @(t,q_A2B) quateqn(q_A2B,w_A2B_B);
```

Solves the ODE from time $t = 0$ to time $t = 10 \, \text{s}$.

```
[t,q_A2B] = ode45(dqdt,0:0.01:10,q_A2B_0);
q_A2B = q_A2B';
```

Gets the rotation matrices at each solution time from the quaternions.

```
R_A2B = zeros(3,3,length(t));
for i = 1:length(t)
```

```
      R_A2B(:,:,i) = quat2mat(q_A2B(:,i));
end
```

Derivative of the rotation matrix (numerical).

Extracting the (1,2) element,

```
R11 = R_A2B(1,2,:);
R11 = R11(:);
```

Evaluating the derivative numerically,

```
dR11dt_numerical = diff(R11)./diff(t);
```

Derivative of the rotation matrix (analytical).

Skew-symmetric form of the angular velocity.

```
W = vec2skew(w_A2B_B);
```

Time derivative of the rotation matrix at each solution time.

```
dRdt = zeros(size(R_A2B));
for i = 1:length(t)
    dRdt(:,:,i) = -W*R_A2B(:,:,i);
end
```

Time derivative of the (1,2) element of the rotation matrix.

```
dR11dt_analytical = dRdt(1,2,:);
dR11dt_analytical = dR11dt_analytical(:);
```

Comparison plot.

Plotting both the numerical and analytical results,

```matlab
figure;
hold on;
plot(t,dR11dt_analytical,'LineWidth',1.5);
plot(t(1:end-1),dR11dt_numerical,'k:','LineWidth',1.5);
hold off;
grid on;
xlabel('$t\;[\mathrm{s}]$','Interpreter','latex','FontSize',18);
ylabel('$dR_{12}/dt\;[\mathrm{1/s}]$','Interpreter','latex','FontSize',18);
legend('analytical','numerical','Interpreter','latex','FontSize',14);
```
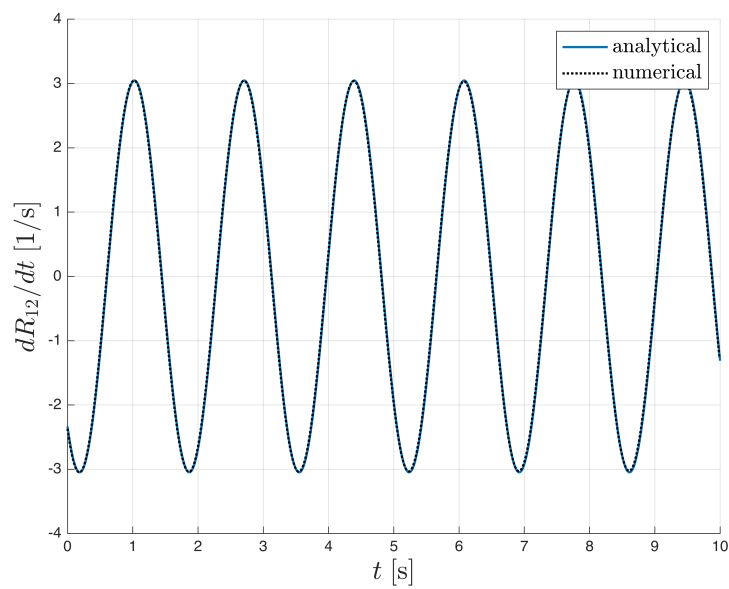
B.4   Derivation: Quaternion Kinematic Equation

derivation_quaternion_kinematic_equation.mlx

Aerospace Simulation Toolbox

Scalar form of the quaternion kinematic equation.

- • Author: Tamas Kis
- • Last Update: 2023-05-09

```matlab
% clears Workspace and Command Window, closes all figures
clear; clc; close all;
```

Scalar form of the quaternion kinematic equation.

Initialize the symbolic variables.

```matlab
syms q_0 q_1 q_2 q_3 p q r;
```

Body rate matrix.

```matlab
Omega = [0  -p  -q  -r;
         p   0   r  -q;
         q  -r   0   p;
         r   q  -p   0];
```

Angular velocity of body frame with respect to the world frame, expressed in the body frame.

```matlab
omega = [p;
         q;
         r];
```

Quaternion from the world frame to the body frame.

```matlab
q = [q_0;
     q_1;
     q_2;
     q_3];
```

Quaternion kinematic equation.

```matlab
dqdt = (1/2)*Omega*q
```

```
dqdt =
```

1

$$
\begin{pmatrix}
-\dfrac{p\,q_1}{2} - \dfrac{q\,q_2}{2} - \dfrac{q_3\,r}{2} \\[2ex]
\dfrac{p\,q_0}{2} - \dfrac{q\,q_3}{2} + \dfrac{q_2\,r}{2} \\[2ex]
\dfrac{p\,q_3}{2} + \dfrac{q\,q_0}{2} - \dfrac{q_1\,r}{2} \\[2ex]
\dfrac{q\,q_1}{2} - \dfrac{p\,q_2}{2} + \dfrac{q_0\,r}{2}
\end{pmatrix}
$$

2

C

Constants

C.1 Physical Constants

Parameter	Description	Value	Units	Source(s)
c	speed of light	299 792 458	$\mathrm{m/s}$	[49, p. 127]
$P_\odot$	solar radiation pressure	4.56×10^{-6}	$\mathrm{N/m^2}$	Section 12.4

C.2 **Earth**

Parameter	Description	Value	Units	Source
AU	astronomical unit	149597870700	m	IAU2012[1]
$\mu_\oplus$	Earth gravitational parameter	$3.986004415 \times 10^{14}$	m^3/s^2	GGM05S[2]
$R_\oplus$	Earth mean equatorial radius	6378136.3	m	GGM05S[3]
$e_\oplus$	Earth (first) eccentricity	0.081819190842622	-	WGS 84[4]
f	Earth flattening	1/298.257223563	-	WGS 84[5]
$\omega_\oplus$	Earth rotational speed	$7.292115146706979 \times 10^{-5}$	rad/s	IAU2000[6]
J_2	2nd degree zonal potential coefficient	0.001082635819197	-	GGM05S[7]
ρ_0	atmospheric density at sea level	1.225	kg/m^3	U.S. Standard Atmosphere, 1976[8]

---

[1] [76]

[2] From `GGM05S.ICGEM.gz` [34]. Also the value listed in [103, p. 1041].

[3] From `GGM05S.ICGEM.gz` [34]. Also the value listed in [103, p. 1041].

[4] [25, p. 3-7]

[5] [25, p. 3-2]

[6] [54, p. 12] and [18, p. 4] give $\omega_\oplus = 1.00273781191135448$ rev/d. Converting this to rad/s,

$$\omega_\oplus = \left(\frac{1.00273781191135448 \text{ rev}}{1 \text{ d}} \right) \left(\frac{2\pi \text{ rad}}{1 \text{ rev}} \right) \left(\frac{1 \text{ d}}{(2400 \text{ s})(24 \text{ h})} \right) = 7.292115146706980 \times 10^{-5} \text{ rad/s}$$

This is very similar to the value of $\omega_\oplus = 7.292115146706979 \times 10^{-5}$ rad/s given by [103, p. 222] (the one digit offset is likely due to numerical issues/rounding). We use the value from [103, p. 222] in the table.

[7] Found using $J_2 = -C_{2,0}$ (Eq. (11.6)). $C_{2,0}$ was found by denormalizing $\overline{C}_{2,0}$ using Algorithm 123. $\overline{C}_{2,0}$ was taken from [34].

[8] [101, p. 20]

C.3 Celestial Bodies

Celestial Body	Gravitational Parameter $[\mathrm{m}^3/s^2]$	Source(s)
Sun (☉)	$1.32712440018 \times 10^{20}$	[7]
Moon (☾)	$4.902800118 \times 10^{12}$	[7]
Mercury (☿)	$2.2031868551 \times 10^{13}$	[7]
Venus (♀)	$3.24858592000 \times 10^{14}$	[7]
Mars system (♂)	$4.2828375816 \times 10^{13}$	[7]
Jupiter system (♃)	$1.26712764100000 \times 10^{17}$	[7]
Saturn system (♄)	$3.7940584841800 \times 10^{16}$	[7]
Uranus system (♅)	$5.794556400000 \times 10^{15}$	[7]
Neptune system (♆)	$6.836527100580 \times 10^{15}$	[7]
Pluto system (♇)	$9.75500000 \times 10^{11}$	[7]

C.4 GNSS

Parameter	Description	Value	Units	Source
f_{L1}	GPS L1 signal frequency	1575.42	MHz	IS-GPS-200M[9]

---

[9] [62]

Bibliography

[1] *n-body problem*. Wikipedia. Accessed: June 4, 2023. URL: https://en.wikipedia.org/wiki/N-body_problem.

[2] *Active and passive transformation*. Wikipedia. Accessed: February 18, 2023. URL: https://en.wikipedia.org/wiki/Active_and_passive_transformation.

[3] *Aircraft flight dynamics*. Wikipedia. Accessed: September 19, 2023. URL: https://en.wikipedia.org/wiki/Aircraft_flight_dynamics.

[4] Kyle T. Alfriend et al. *Spacecraft Formation Flying*. Oxford, UK: Elsevier, 2010.

[5] *Angle between 2 quaternions*. MathWorks. Accessed: May 7, 2023. URL: https://www.mathworks.com/matlabcentral/answers/415936-angle-between-2-quaternions?s_tid=answers_rc1-2_p2_MLT.

[6] *Angular velocity*. Wikipedia. Accessed: March 25, 2023. URL: https://en.wikipedia.org/wiki/Angular_velocity.

[7] *Astrodynamic Parameters*. NASA Jet Propulsion Laboratory. Accessed: February 5, 2022. URL: https://ssd.jpl.nasa.gov/astro_par.html.

[8] *Astronomical symbols*. Wikipedia. Accessed: August 19, 2023. URL: https://en.wikipedia.org/wiki/Astronomical_symbols.

[9] *Attitude Transformations*. VectorNav. Accessed: April 23, 2023. URL: https://www.vectornav.com/resources/inertial-navigation-primer/math-fundamentals/math-attitudetran.

[10] *Axis-angle representation*. Wikipedia. Accessed: April 24, 2023. URL: https://en.wikipedia.org/wiki/Axis-angle_representation.

[11] Martin Baker. *Maths - Conversion Axis-Angle to Euler*. EuclidianSpace. Accessed: April 25, 2023. URL: https://www.euclideanspace.com/maths/geometry/rotations/conversions/angleToEuler/index.htm.

[12] *Basis (linear algebra)*. Wikipedia. Accessed: February 22, 2023. URL: https://en.wikipedia.org/wiki/Basis_(linear_algebra).

[13] Roger R. Bate, Donald D. Mueller, and Jerry E. White. *Fundamentals of Astrodynamics*. 1st. New York, NY: Dover Publications, 1971.

[14] Ferdinand P. Beer, Jr. E. Russell Johnston, and Phillip J. Cornwell. *Vector Mechanics for Engineers*. 10th. New York, NY: McGraw Hill, 2013.

[15] Rebecca M. Brannon. *Lecture 12: Three-Dimensional Rotation Matrices*. Physics 216 Lecture Notes (UC Santa Cruz). 2012. URL: http://scipp.ucsc.edu/~haber/ph216/rotation_12.pdf.

[16] Rebecca M. Brannon. *Rotation: A review of theorems involving proper orthogonal matrices referenced to three-dimensional physical space*. Sandia National Laboratories. 2002. URL: https://my.mech.utah.edu/~brannon/public/rotation.pdf.

[17] Matthew Brett. *quat2axang*. Transforms3d. Accessed: May 1, 2023. URL: https://github.com/matthew-brett/transforms3d/blob/main/transforms3d/quaternions.py.

[18]   Nicole Capitaine and Patrick T. Wallace. *Implementation of the IAU2000 definition of UT1 in astronomy*. Accessed: January 6, 2022. URL: https://www.atnf.csiro.au/iau-comm31/pdf/2009_IAUGA_JD6/JD06_capitaine_wallace.pdf.

[19]   J. O. Cappellari, C. E. Velez, and A. J. Fuchs. *Mathematical Theory of the Goddard Trajectory Determination System*. Tech. rep. GSFC X-582-76-77. Greenbelt, MD: Goddard Space Flight Center, Apr. 1976. URL: https://ntrs.nasa.gov/api/citations/19760017203/downloads/19760017203.pdf.

[20]   *Cartesian coordinate system*. Wikipedia. Accessed: October 25, 2023. URL: https://en.wikipedia.org/wiki/Cartesian_coordinate_system.

[21]   *Coordinate vector*. Wikipedia. Accessed: February 20, 2023. URL: https://en.wikipedia.org/wiki/Coordinate_vector.

[22]   *Cross Product*. Wikipedia. Accessed: February 27, 2022. URL: https://en.wikipedia.org/wiki/Cross_product.

[23]   Howard D. Curtis. *Orbital Mechanics for Engineering Students*. 3rd. Oxford, UK: Butterworth-Heinemann, 2014.

[24]   Neil Dantam. *Quaternion Computation*. 2014. URL: http://www.neil.dantam.name/note/dantam-quaternion.pdf.

[25]   *Department of Defense World Geodetic System 1984*. Tech. rep. NIMA TR8350.2. National Imagery and Mapping Agency, 2004. URL: https://apps.dtic.mil/sti/pdfs/AD1000581.pdf.

[26]   *Description of Files Related to Using the EGM2008 Global Gravitational Model to Compute Geoid Undulations with Respect to WGS 84*. NGA Office of Geomatics. file name in download: README$_W GS84_2.pdf$. URL: https://earth-info.nga.mil/php/download.php?file=egm-08spherical.

[27]   *det(A) = 1 implies A is orthogonal*. Stack Exchange. Accessed: February 19, 2023. URL: https://math.stackexchange.com/questions/2449077/textdeta-1-implies-a-is-orthogonal.

[28]   James Diebel. *Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors*. Stanford University. Accessed: April 29, 2023. 2006. URL: https://www.astro.rug.nl/software/kapteyn-beta/_downloads/attitude.pdf.

[29]   *Dynamical time scale*. Wikipedia. Accessed: April 1, 2023. URL: https://en.wikipedia.org/wiki/Dynamical_time_scale.

[30]   Randy A. Eckman, Aaron J. Brown, and Daniel R. Adamo. "Normalization of Gravitational Acceleration Models". In: *NASA* (2011). URL: https://ntrs.nasa.gov/api/citations/20110023121/downloads/20110023121.pdf.

[31]   *Ellipse*. Wikipedia. Accessed: August 19, 2023. URL: https://en.wikipedia.org/wiki/Ellipse.

[32]   *Euclidean space*. Wikipedia. Accessed: October 25, 2023. URL: https://en.wikipedia.org/wiki/Euclidean_space.

[33]   *Euler Angles*. Academic Flight. Accessed: February 18, 2023. URL: https://academicflight.com/articles/kinematics/rotation-formalisms/euler-angles/.

[34]   *GGM05S.ICGEM.gz*. Center for Space Research (The University of Texas at Austin). Accessed: February 5, 2022. URL: http://download.csr.utexas.edu/pub/grace/GGM05/.

[35]   *Gimbal lock*. Wikipedia. Accessed: April 22, 2023. URL: https://en.wikipedia.org/wiki/Gimbal_lock.

[36]   *Gravity Model Files*. a.i. solutions. Accessed: January 6, 2022. URL: https://ai-solutions.com/_help_Files/gravity_model_files.htm.

[37]   Donald T. Greenwood. *Principles of Dynamics*. 2nd. Upper Saddle River, NJ: Prentice-Hall, 1988.

[38]   *Gregorian calendar*. Wikipedia. Accessed: January 7, 2022. URL: https://en.wikipedia.org/wiki/Gregorian_calendar.

[39]   Jozef C. van der Ha and Malcolm D. Shuster. "A Tutorial on Vectors and Attitude". In: *IEEE Control Systems Magazine* 29.2 (2009), pp. 94–107. DOI: 10.1109/MCS.2008.929426. URL: http://www.malcolmdshuster.com/Pubp_018_073y_J_CSM_Vecs&Att_MDS.pdf.

[40] *How to calculate the flight path angle, γ, from a state vector?* Stack Exchange. Accessed: August 31, 2023. URL: `https://space.stackexchange.com/questions/28426/how-to-calculate-the-flight-path-angle-%CE%B3-from-a-state-vector`.

[41] *IERS Bulletins*. International Earth Rotation and Reference Systems Service. Accessed: January 22, 2022. URL: `https://www.iers.org/IERS/EN/Publications/Bulletins/bulletins.html`.

[42] *Inertial frame of reference*. Wikipedia. Accessed: June 22, 2020. URL: `https://en.wikipedia.org/wiki/Inertial_frame_of_reference`.

[43] *Kepler's laws of planetary motion*. Wikipedia. Accessed: September 12, 2023. URL: `https://en.wikipedia.org/wiki/Kepler's_laws_of_planetary_motion`.

[44] Axel Kielhorn. *The wasysym macro package for LATEX*. CTAN. Accessed: August 19, 2023. URL: `https://ctan.mirrors.hoobly.com/macros/latex/contrib/wasysym/wasysym.pdf`.

[45] Tamas Kis. *Algorithms for Interpolation*. 2022. URL: `https://tamaskis.github.io/files/Algorithms_for_Interpolation.pdf`.

[46] Andrea L'Afflitto. *L'Afflitto - A Mathematical Perspective on Flight Dynamics and Control*. Springer, 2017.

[47] A. K. Lal. *Ordered Basis*. 2007. URL: `https://archive.nptel.ac.in/content/storage2/courses/122104018/node41.html`.

[48] *Laplace-Runge-Lenz vector*. Wikipedia. Accessed: August 19, 2023. URL: `https://en.wikipedia.org/wiki/Laplace%E2%80%93Runge%E2%80%93Lenz_vector`.

[49] *Le Système international dunités (The International System of Units)*. Tech. rep. 9th Edition. Bureau International des Poids et Mesures, 2019. URL: `https://www.bipm.org/en/publications/si-brochure`.

[50] *Leap_Second_History.dat*. l'Observatoire de Paris Earth Orientation Center. Accessed: January 22, 2022. URL: `https://hpiers.obspm.fr/eoppc/bul/bulc/Leap_Second_History.dat`.

[51] Steven J. Leon. *Linear Algebra with Applications*. 9th. London: Pearson, 2015.

[52] *Lunar Gravity Field: GRGM1200A*. NASA Goddard Space Flight Center. Accessed: June 24, 2022. URL: `https://pgda.gsfc.nasa.gov/products/50`.

[53] Jaakko Mäkinen. "The permanent tide and the International Height Reference Frame IHRF". In: *Journal of Geodesy* 95.106 (2021). DOI: `10.1007/s00190-021-01541-5`.

[54] Dennis D. McCarthy and Nicole Capitaine. "Practical Consequences of Resolution B1.6, Resolution B1.7, Resolution B1.8". In: *Proceedings of the IERS Workshop on the Implementation of the New IAU Resolutions* IERS Technical Note No. 29 (2002). URL: `https://www.iers.org/SharedDocs/Publikationen/EN/IERS/Publications/tn/TechnNote29/tn29_009.pdf?__blob=publicationFile&v=1`.

[55] *Mechanical energy*. Wikipedia. Accessed: June 5, 2023. URL: `https://en.wikipedia.org/wiki/Mechanical_energy`.

[56] *Minute and second of arc*. Wikipedia. Accessed: September 27, 2021. URL: `https://en.wikipedia.org/wiki/Minute_and_second_of_arc`.

[57] *mjuliandate*. MathWorks. Accessed: April 5, 2023. URL: `https://www.mathworks.com/help/aerotbx/ug/mjuliandate.html`.

[58] *Moment of inertia*. Wikipedia. Accessed: February 18, 2023. URL: `https://en.wikipedia.org/wiki/Moment_of_inertia`.

[59] *Momentum*. Wikipedia. Accessed: October 20, 2023. URL: `https://en.wikipedia.org/wiki/Momentum`.

[60] Oliver Montenbruck and Eberhard Gill. *Satellite Orbits – Models, Methods, Applications*. 4th. Berlin, Heidelberg: Springer-Verlag, 2012.

[61] Oliver Montenbruck and Thomas Pfleger. *Astronomy on the Personal Computer*. 4th. Berlin, Heidelberg, New York: Springer-Verlag, 2000.

[62] *NAVSTAR GPS Space Segment/Navigation User Interfaces*. Tech. rep. IS-GPS-200M. May 2021. URL: `https://www.gps.gov/technical/icwg/IS-GPS-200M.pdf`.

[63] *Newton's laws of motion*. Wikipedia. Accessed: October 20, 2023. URL: https://en.wikipedia.org/wiki/Newton's_laws_of_motion.

[64] *Newtonian constant of gravitation*. NIST. Accessed: June 4, 2023. URL: https://physics.nist.gov/cgi-bin/cuu/Value?bg.

[65] *Newtonian constant of gravitation*. Wikipedia. Accessed: June 4, 2023. URL: https://en.wikipedia.org/wiki/Newton's_law_of_universal_gravitation.

[66] *Numerical precision of arctan function*. Stack Exchange. Accessed: May 8, 2023. URL: https://math.stackexchange.com/questions/1335090/numerical-precision-of-arctan-function.

[67] Nikolaos K. Pavlis et al. "The development and evaluation of the Earth GravitationalModel 2008 (EGM2008)". In: *Journal of Geophysical Research* 118 (2012). DOI: 10.1029/2011JB008916.

[68] Gérard Petit and Brian Luzum. "IERS Conventions (2010)". In: *International Earth Rotation and Reference Systems Service (IERS)* IERS Technical Note No. 36 (2010). URL: https://www.iers.org/SharedDocs/Publikationen/EN/IERS/Publications/tn/TechnNote36/tn36.pdf?__blob=publicationFile&v=1.

[69] *Point particle*. Wikipedia. Accessed: October 20, 2023. URL: https://en.wikipedia.org/wiki/Point_particle.

[70] *Principal Rotation Vector (Axis-Angle Formalism)*. Academic Flight. Accessed: April 23, 2023. URL: https://academicflight.com/articles/kinematics/rotation-formalisms/principal-rotation-vector/.

[71] *quat2dcm*. MathWorks. Accessed: April 29, 2023. URL: https://www.mathworks.com/help/aerotbx/ug/quat2dcm.html.

[72] *Quaternions and spatial rotations*. Wikipedia. Accessed: April 22, 2023. URL: https://en.wikipedia.org/wiki/Quaternion.

[73] *Quaternions and spatial rotations*. Wikipedia. Accessed: April 22, 2023. URL: https://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation.

[74] *quatmultiply*. MathWorks. Accessed: May 5, 2023. URL: https://www.mathworks.com/help/aerotbx/ug/quatmultiply.html.

[75] Anil V. Rao. *Rao - Orbital Mechanics: An Introduction*. Gainesville, FL: University of Florida, 2021. URL: https://www.anilvrao.com/resources/EAS4510-Course-Notes.pdf.

[76] "Resolution B2 on the re-definition of the astronomical unit of length". In: *XXVIII General Assembly of International Astronomical Union* (2012). URL: https://www.iau.org/static/resolutions/IAU2012_English.pdf.

[77] *Rotating reference frame*. Wikipedia. Accessed: June 19, 2023. URL: https://en.wikipedia.org/wiki/Rotating_reference_frame.

[78] *Rotation matrix*. Wikipedia. Accessed: April 24, 2023. URL: https://en.wikipedia.org/wiki/Rotation_matrix.

[79] *Rotation Matrix / Direction Cosine Matrix (DCM)*. Academic Flight. Accessed: June 18, 2023. URL: https://academicflight.com/articles/kinematics/rotation-formalisms/rotation-matrix/.

[80] *Rotations with Quaternions*. Academic Flight. Accessed: April 23, 2023. URL: https://academicflight.com/articles/kinematics/rotation-formalisms/quaternions/.

[81] Soheil Sarabandi and Federico Thomas. "A Survey on the Computation of Quaternions From Rotation Matrices". In: *Journal of Mechanisms and Robotics* 11.2 (2019). DOI: 10.1115/1.4041889. URL: https://upcommons.upc.edu/bitstream/handle/2117/178326/2083-A-Survey-on-the-Computation-of-Quaternions-from-Rotation-Matrices.pdf?sequence=1.

[82] Soheil Sarabandi and Federico Thomas. "Accurate Computation of Quaternions from Rotation Matrices". In: *Advances in Robot Kinematics*. Springer Proceedings in Advanced Robotics. Springer, 2019, pp. 39–46. DOI: 10.1007/978-3-319-93188-3_5. URL: http://www.iri.upc.edu/files/scidoc/2068-Accurate-Computation-of-Quaternions-from-Rotation-Matrices.pdf.

[83]   Lester W. Schmerr. *Engineering Dynamics 2.0*. Switzerland: Springer Nature, 2019.

[84]   Ken Shoemake. "Animating Rotations with Quaternion Curves". In: *SIGGRAPH '85: Proceedings of the 12 annual conference on Computer graphics and interactive techniques*. Vol. 19. 3. ACM, 1985, pp. 245–254. DOI: 10.1145/325334.325242.

[85]   *Show that any orthogonal matrix has a determine 1 or -1*. Stack Exchange. Accessed: February 19, 2023. URL: https://math.stackexchange.com/questions/1172802/show-that-any-orthogonal-matrix-has-determinant-1-or-1.

[86]   Malcolm D. Shuster. *A Tutorial on Attitude Kinematics*. 2008. URL: http://www.malcolmdshuster.com/Pubp_022_022y_J_AttKin_MDS.pdf.

[87]   *Skew-symmetric matrix*. Wikipedia. Accessed: May 7, 2023. URL: https://en.wikipedia.org/wiki/Skew-symmetric_matrix.

[88]   *Slerp*. Wikipedia. Accessed: May 7, 2023. URL: https://en.wikipedia.org/wiki/Slerp.

[89]   Harry Smith. *Axes Transformations*. Aircraft Flight Mechanics. Accessed: February 18, 2023. URL: https://aircraftflightmechanics.com/EoMs/EulerTransforms.html.

[90]   Joan Solà. "Quaternion kinematics for the error-state Kalman filter". In: *arXiv* (2017). DOI: 10.48550/arXiv.1711.02508.

[91]   Brian L. Stevens, Frank L. Lewis, and Eric N. Johnson. *Aircraft Control and Simulation*. 3rd. Hoboken, NJ: John Wiley & Sons, 2016.

[92]   Iain W. Stewart. *Advanced Classical Mechanics*. MIT OpenCourseWare, 2016. URL: https://ocw.mit.edu/courses/physics/8-09-classical-mechanics-iii-fall-2014/lecture-notes/MIT8_09F14_full.pdf.

[93]   James Stewart. "Chapter 12: Vectors and the Geometry of Space". In: *Calculus*. 8th. Boston, MA: Cengage Learning, 2015, pp. 831–885.

[94]   *Tensor*. Wikipedia. Accessed: February 18, 2023. URL: https://en.wikipedia.org/wiki/Tensor.

[95]   Ashish Tewari. *Atmospheric and Space Flight Dynamics*. Boston: Birkhäuser, 2007.

[96]   *The Unnecessarily Short Ways To Do a Quaternion Slerp*. Magnum Engine. Accessed: May 8, 2023. URL: https://blog.magnum.graphics/backstage/the-unnecessarily-short-ways-to-do-a-quaternion-slerp/.

[97]   *Three-dimensional space*. Wikipedia. Accessed: October 25, 2023. URL: https://en.wikipedia.org/wiki/Three-dimensional_space.

[98]   *Transport theorem*. Wikipedia. Accessed: March 28, 2023. URL: https://en.wikipedia.org/wiki/Transport_theorem.

[99]   *Triple product*. Wikipedia. Accessed: August 19, 2023. URL: https://en.wikipedia.org/wiki/Triple_product.

[100] *Two-body problem*. Wikipedia. Accessed: June 4, 2023. URL: https://en.wikipedia.org/wiki/Two-body_problem.

[101] *U.S. Standard Atmosphere, 1976*. Tech. rep. NOAA-S/T-76-1562. National Oceanic and Atmospheric Administration (NOAA), 1976. URL: https://ntrs.nasa.gov/citations/19770009539.

[102] *Universal time*. Wikipedia. Accessed: April 1, 2023. URL: https://en.wikipedia.org/wiki/Universal_time.

[103] David A. Vallado. *Fundamentals of Astrodynamics and Applications*. 4th. Hawthorne, CA: Microcosm Press, 2013.

[104] *Vis-viva equation*. Wikipedia. Accessed: August 20, 2023. URL: https://en.wikipedia.org/wiki/Vis-viva_equation.

[105] James R. Wertz. *Spacecraft Attitude Determination and Control*. Dordrecht, Holland: D. Reidel Publishing Company, 1978.

[106] *What is the range of the zenith angle between velocity and position vectors?* Stack Exchange. Accessed: September 19, 2023. URL: https://astronomy.stackexchange.com/questions/28865/what-is-the-range-of-the-zenith-angle-between-velocity-and-position-vectors.

[107] *Why is the matrix product of 2 orthogonal matrices also an orthogonal matrix?* Stack Exchange. Accessed: July 5, 2020. URL: https://math.stackexchange.com/questions/1416726/why-is-the-matrix-product-of-2-orthogonal-matrices-also-an-orthogonal-matrix.

[108] *Work (physics)*. Wikipedia. Accessed: June 5, 2023. URL: https://en.wikipedia.org/wiki/Work_(physics).

[109] *WORLD GEODETIC SYSTEM 1984 (WGS 84)*. NGA Office of Geomatics. Accessed: June 25, 2022. URL: https://earth-info.nga.mil/index.php?dir=wgs84&action=wgs84.

[110] Shiyu Zhao. "Time Derivative of Rotation Matrices: A Tutorial". In: *arXiv* (2016). DOI: 10.48550/arXiv.1609.06088.