Attitude Determination

ASEN 5010

Dr. Vishala Arya
vishala.arya@colorado.edu

University of Colorado
Boulder

Introduction

- Attitude determination is broken up into two areas
  - **Static attitude determination**: All measurements are taken at the same time. Using this snap shot in time concept, the problem becomes up of optimally solving the geometry of the measurements
  - **Dynamic attitude determination**: Here measurements are taken over time. This is a much harder problem, in that attitude measurements are taken over time, along with some gyro (rotation rate) measurements, which then need to be optimally blended together (Kalman filter).

Basic Concept

- Consider the 2D attitude problem. How many direction measurements (unit direction vectors) does it take to determine your heading?

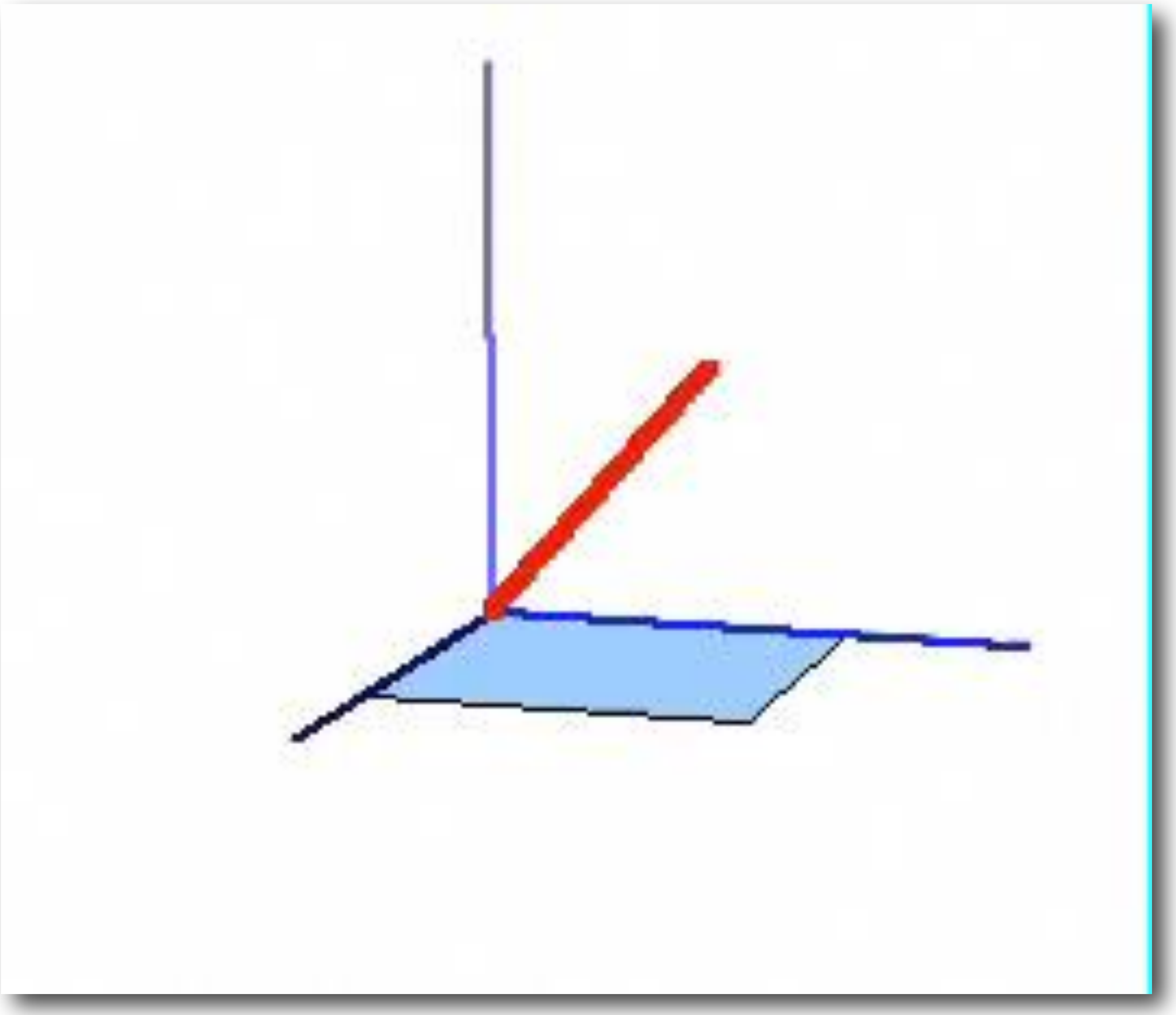
Answer: You only need one direction measurement for the 2D case.

Explanation: Headings in a 2D environment is a 1D measure. The unit direction vector (with the unit length constraint) provides all the required information.

- Next, let us consider the three dimensional orientation measurement. How many observation vectors (unit direction vectors) are required here?

Answer: You will need a minimum of two observation vectors.

Explanation: With only one measurement, you cannot sense rotations about this axis. Measuring a second direction will fix the complete three dimension orientation in space.



- To determine attitude, we assume you already know the inertial direction to certain objects (sun, Earth, magnetic field direction, stars, moon, etc.)
- Assume the sun direction is given by $\hat{\boldsymbol{s}}$ and the local magnetic field direction is given by $\hat{\boldsymbol{m}}$
- If the vehicle has sensors on board that measure these directions, then these unit vectors are measured with components taken in the vehicle fixed body frame B.

Measured: ${}^{\mathcal{B}}\hat{\boldsymbol{m}}$ ${}^{\mathcal{B}}\hat{\boldsymbol{s}}$

Given: ${}^{\mathcal{N}}\hat{\boldsymbol{m}}$ ${}^{\mathcal{N}}\hat{\boldsymbol{s}}$

Mapping:

$$ {}^{\mathcal{B}}\hat{\boldsymbol{m}} = [\bar{B}N]\,{}^{\mathcal{N}}\hat{\boldsymbol{m}} $$

$$ {}^{\mathcal{B}}\hat{\boldsymbol{s}} = [\bar{B}N]\,{}^{\mathcal{N}}\hat{\boldsymbol{s}} $$

Challenge: How do we find $[BN]$?

Under or Over?

- Note that each observation vector (unit direction vector) contains two independent degrees of freedom.
- The 3D attitude problem is a three-degree of freedom problem.
- Thus, by measuring two observation directions, the attitude determination problem is always an over-determined problem!

Deterministic Attitude Estimation

Vector Triad Method

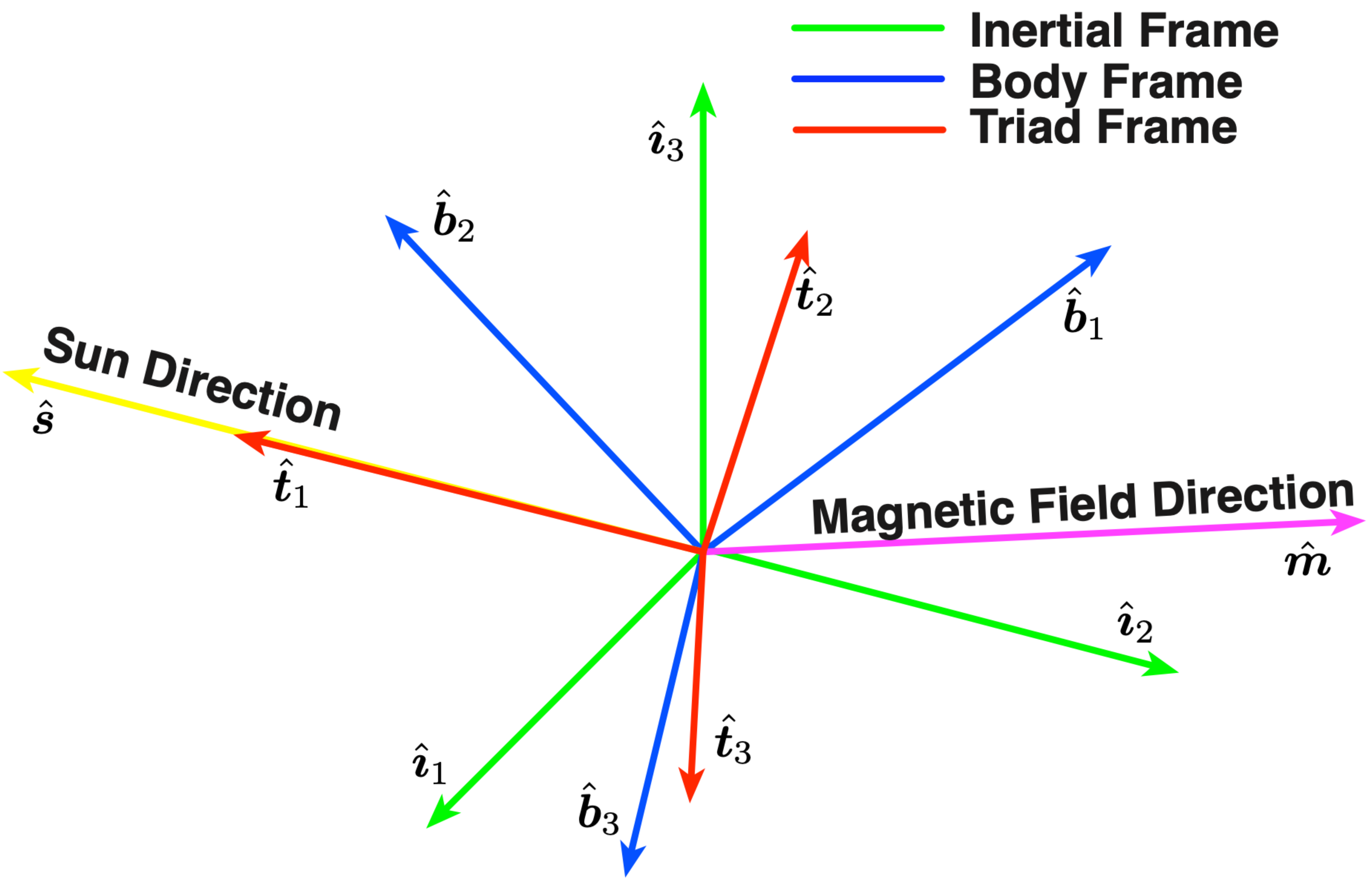
- To determine the desired [*BI*] matrix, we first introduce the triad coordinate frame *T*.

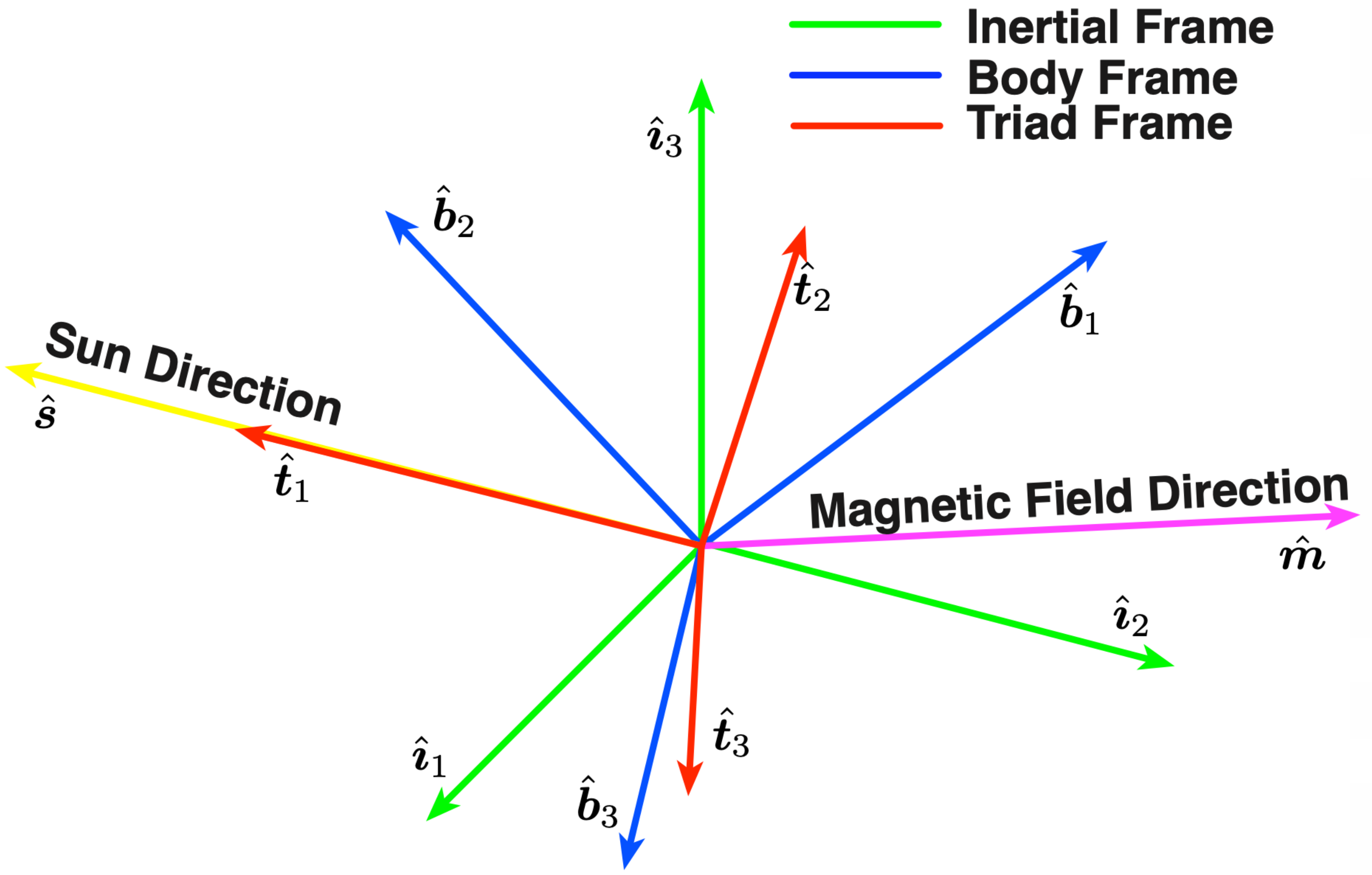
Assume: $\hat{\boldsymbol{t}}_1 = \hat{\boldsymbol{s}}$

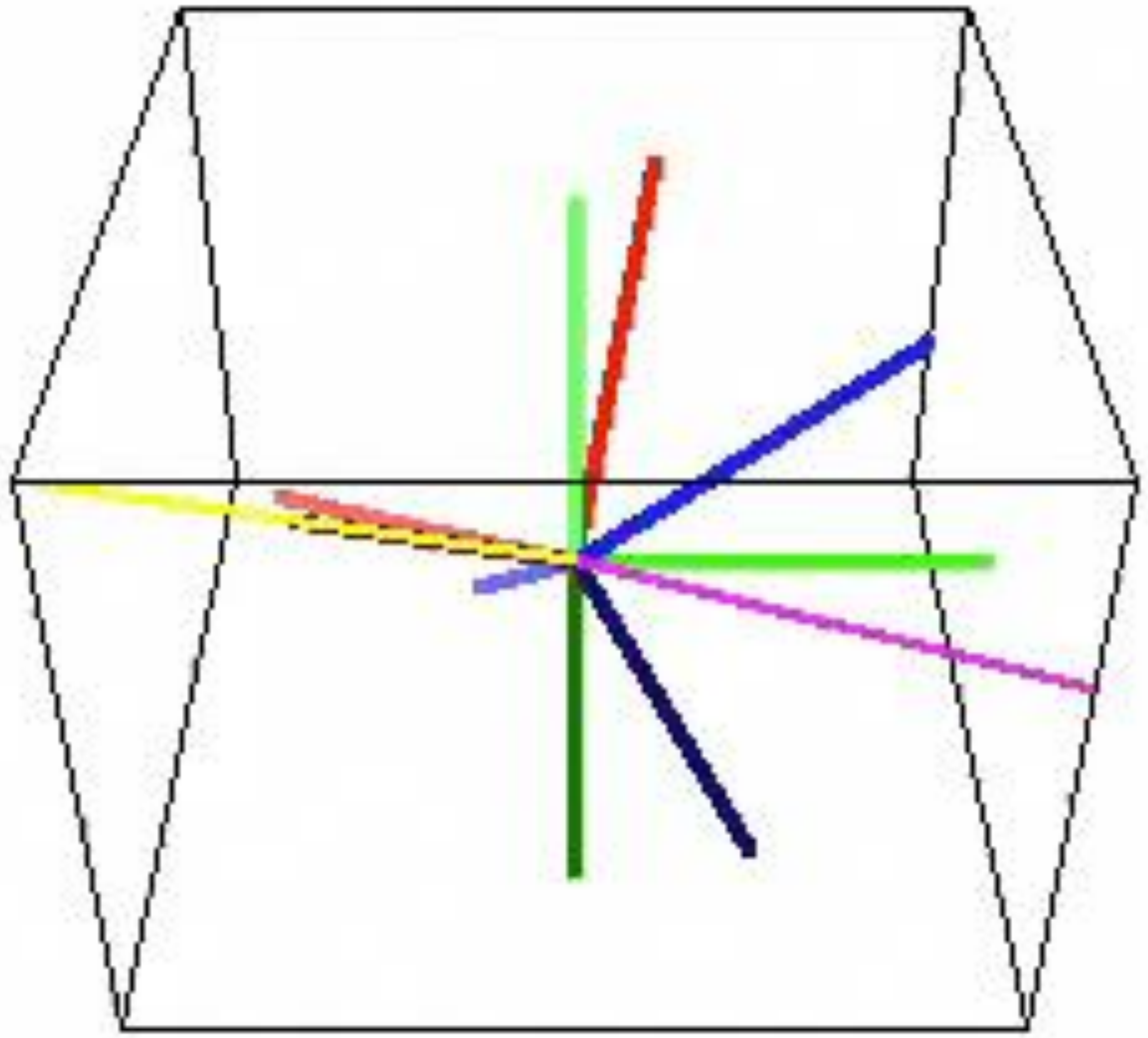
Then define:

$$\hat{\boldsymbol{t}}_2 = \frac{\hat{\boldsymbol{s}} \times \hat{\boldsymbol{m}}}{|\hat{\boldsymbol{s}} \times \hat{\boldsymbol{m}}|}$$

$$\hat{\boldsymbol{t}}_3 = \hat{\boldsymbol{t}}_1 \times \hat{\boldsymbol{t}}_2$$







3D Illustration of Triad Coordinate Frame

- We can compute the *T* frame direction axes using both *B* and I frame components using

$$ {}^{\mathcal{B}}\hat{\boldsymbol{t}}_1 = {}^{\mathcal{B}}\hat{\boldsymbol{s}} $$
$$ {}^{\mathcal{B}}\hat{\boldsymbol{t}}_2 = \frac{({}^{\mathcal{B}}\hat{\boldsymbol{s}}) \times ({}^{\mathcal{B}}\hat{\boldsymbol{m}})}{|({}^{\mathcal{B}}\hat{\boldsymbol{s}}) \times ({}^{\mathcal{B}}\hat{\boldsymbol{m}})|} $$
$$ {}^{\mathcal{B}}\hat{\boldsymbol{t}}_3 = ({}^{\mathcal{B}}\hat{\boldsymbol{t}}_1) \times ({}^{\mathcal{B}}\hat{\boldsymbol{t}}_2) $$

Body Frame Triad Vectors

$$ {}^{\mathcal{N}}\hat{\boldsymbol{t}}_1 = {}^{\mathcal{N}}\hat{\boldsymbol{s}} $$
$$ {}^{\mathcal{N}}\hat{\boldsymbol{t}}_2 = \frac{({}^{\mathcal{N}}\hat{\boldsymbol{s}}) \times ({}^{\mathcal{N}}\hat{\boldsymbol{m}})}{|({}^{\mathcal{N}}\hat{\boldsymbol{s}}) \times ({}^{\mathcal{N}}\hat{\boldsymbol{m}})|} $$
$$ {}^{\mathcal{N}}\hat{\boldsymbol{t}}_3 = ({}^{\mathcal{N}}\hat{\boldsymbol{t}}_1) \times ({}^{\mathcal{N}}\hat{\boldsymbol{t}}_2) $$

Inertial Frame Triad Vectors

- In the absence of measurement errors, both sets of Triad frame representations should be the same.

- We can write the various rotation matrices as

$$ [\bar{B}T] = \begin{bmatrix} {}^{\mathcal{B}}\hat{\boldsymbol{t}}_1 & {}^{\mathcal{B}}\hat{\boldsymbol{t}}_2 & {}^{\mathcal{B}}\hat{\boldsymbol{t}}_3 \end{bmatrix} \qquad [NT] = \begin{bmatrix} {}^{\mathcal{N}}\hat{\boldsymbol{t}}_1 & {}^{\mathcal{N}}\hat{\boldsymbol{t}}_2 & {}^{\mathcal{N}}\hat{\boldsymbol{t}}_3 \end{bmatrix} $$

- Finally, we can compute the desired DCM matrix using

$$[\bar{B}N] = [\bar{B}T][NT]^T$$

- From the rotation matrix, we can now extract any desired set of attitude coordinates!

- Note that with this method we do not use the full magnetic field direction vector $\hat{\boldsymbol{m}}$ If this measurement were more accurate, then we could modify this method to define $\hat{\boldsymbol{t}}_1 = \hat{\boldsymbol{m}}$ instead.

See Mathematica Solution of Example 3.14

Statistical Attitude Determination

Wahba's Problem

- Assume we have N>1 observation measurements (i.e. measured directions to sun, magnetic field, stars, etc.), and we know the corresponding inertial vector directions. Then we can write attitude determination problem as

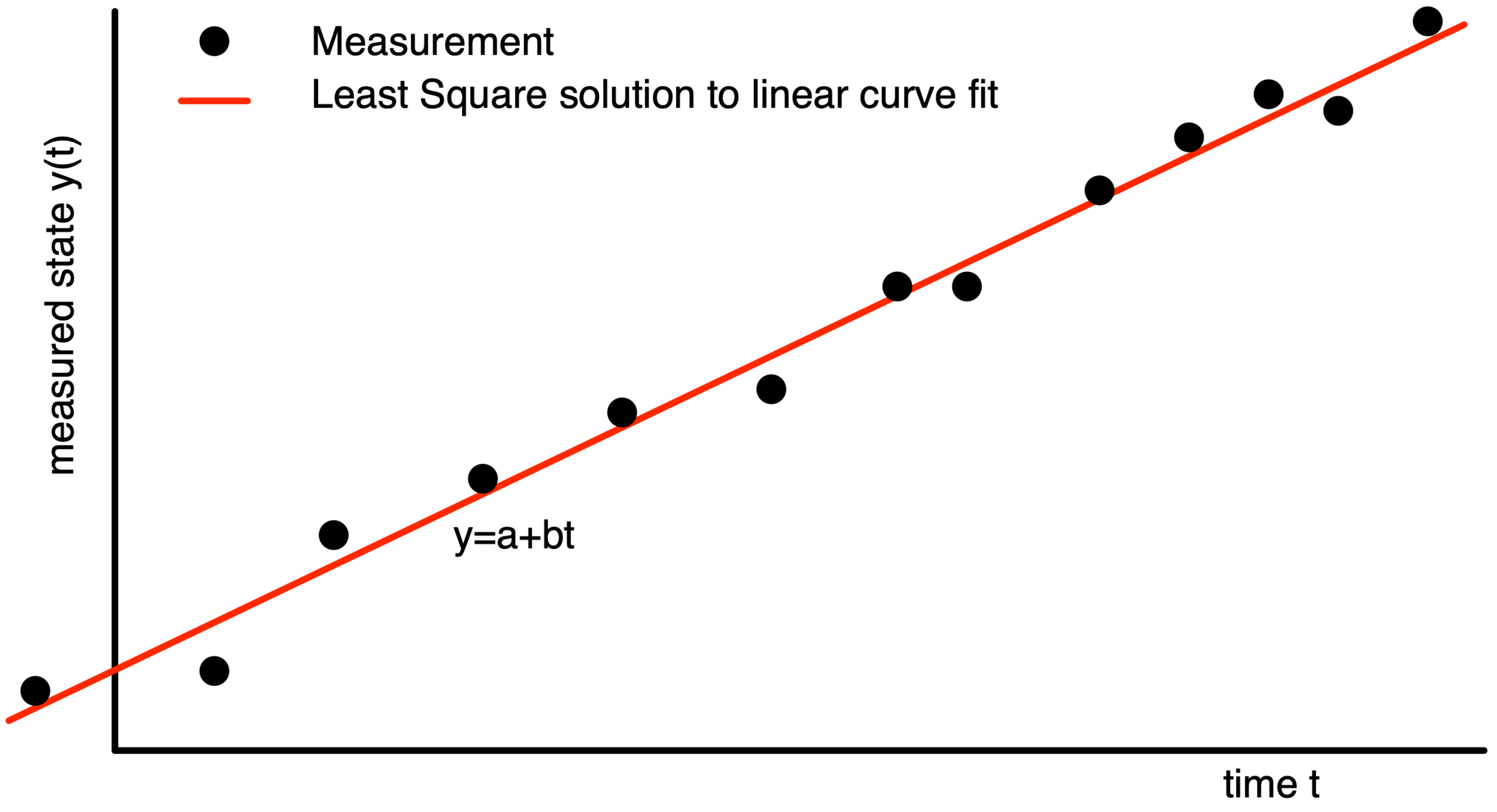
$$ {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k = [\bar{B}N]\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k \qquad \text{for } k = 1, \cdots, N $$

  with the goal to find the rotation matrix [BN] such that the following loss function is minimized:

- If all measurements are perfect, then J = 0.

$$ J([\bar{B}N]) = \frac{1}{2}\sum_{k=1}^{N} w_k \left| {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k - [\bar{B}N]\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k \right|^2 $$

• Think of the cost function *J* as the error of the common least squares curve fitting problem:



Devenport's *q*-Method

- Let the 4-D Euler parameter (quaternion) vector be defined as

$$\bar{\boldsymbol{\beta}} = (\beta_0, \beta_1, \beta_2, \beta_3)^T$$

- The cost function can be rewritten

$$J = \frac{1}{2}\sum_{k=1}^{N} w_k \left({}^{\mathcal{B}}\hat{\boldsymbol{v}}_k - [\bar{B}N]\,{}^{\mathcal{N}}\hat{\boldsymbol{v}}_k\right)^T \left({}^{\mathcal{B}}\hat{\boldsymbol{v}}_k - [\bar{B}N]\,{}^{\mathcal{N}}\hat{\boldsymbol{v}}_k\right)$$

$$J = \frac{1}{2}\sum_{k=1}^{N} w_k \left({}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T\,{}^{\mathcal{B}}\hat{\boldsymbol{v}}_k + {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k^T\,{}^{\mathcal{N}}\hat{\boldsymbol{v}}_k - 2\,{}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T[\bar{B}N]\,{}^{\mathcal{N}}\hat{\boldsymbol{v}}_k\right)$$

$$J = \sum_{k=1}^{N} w_k \left(1 - {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T\,[\bar{B}N]\,{}^{\mathcal{N}}\hat{\boldsymbol{v}}_k\right)$$

- Minimizing *J* is equivalent to maximizing the gain function *g*:

$$g = \sum_{k=1}^{N} w_k \, {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T \; [\bar{B}N] \; {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k$$

- The rotation matrix can be written in terms of Euler parameters as

$$[\bar{B}N] = (\beta_0^2 - \boldsymbol{\epsilon}^T\boldsymbol{\epsilon})[I_{3\times 3}] + 2\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T - 2\beta_0[\tilde{\boldsymbol{\epsilon}}] \qquad \boldsymbol{\epsilon} = (\beta_1, \beta_2, \beta_3)$$

- This allows us to rewrite the gain function *g()* using the 4x4 matrix [*K*]

$$\boxed{g(\bar{\boldsymbol{\beta}}) = \bar{\boldsymbol{\beta}}^T[K]\bar{\boldsymbol{\beta}}}$$

$$[K] = \begin{bmatrix} \sigma & Z^T \\ Z & S - \sigma I_{3\times 3} \end{bmatrix}$$

$$[B] = \sum_{k=1}^{N} w_k \, {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k \, {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k^T$$

$$[S] = [B] + [B]^T$$

$$\sigma = \operatorname{tr}([B])$$

$$[Z] = \begin{bmatrix} B_{23} - B_{32} & B_{31} - B_{13} & B_{12} - B_{21} \end{bmatrix}^T$$

- However, since the Euler parameter vector must abide by the unit length constraint, we cannot solve this gain function directly. Instead, we use Lagrange multipliers to yield a new gain function *g*’

$$g'(\bar{\boldsymbol{\beta}}) = \bar{\boldsymbol{\beta}}^T[K]\bar{\boldsymbol{\beta}} - \lambda(\bar{\boldsymbol{\beta}}^T\bar{\boldsymbol{\beta}} - 1)$$

- We differentiate *g*’ and set it equal to zero to find the extrema point of this function.

$$\frac{\mathrm{d}}{\mathrm{d}\bar{\boldsymbol{\beta}}}(g'(\bar{\boldsymbol{\beta}})) = 2[K]\bar{\boldsymbol{\beta}} - 2\lambda\bar{\boldsymbol{\beta}} = 0 \quad \Rightarrow \quad \boxed{[K]\bar{\boldsymbol{\beta}} = \lambda\bar{\boldsymbol{\beta}}}$$

- Clearly the desired Euler parameter vector is the eigenvector of the [*K*] matrix.
- To maximize the gain function, we need to choose the largest eigenvalue of the [*K*] matrix.

$$g(\bar{\boldsymbol{\beta}}) = \bar{\boldsymbol{\beta}}^T[K]\bar{\boldsymbol{\beta}} = \bar{\boldsymbol{\beta}}^T\lambda\bar{\boldsymbol{\beta}} = \lambda\bar{\boldsymbol{\beta}}^T\bar{\boldsymbol{\beta}} = \lambda$$

- In summary, to use the q-Method, we must
  - Compute the 4x4 matrix $[K]$
  - Find the eigenvalue and eigenvector of the $[K]$ matrix
  - Choose the largest eigenvalue and associated eigenvector.
  - This eigenvector is the Euler parameter vector

- Note that solving this eigenvalue, eigenvector problem is numerically rather intensive for real-time applications.

See Mathematica Solution of Example 3.15

QUEST

- Recall the cost function *J* and the gain function *g*

$$J = \sum_{k=1}^{N} w_k \left(1 - {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T \, [\bar{B}N] \, {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k\right)$$

$$g = \sum_{k=1}^{N} w_k \, {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T \, [\bar{B}N] \, {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k$$

- Further, we found that the optimal *g()* will be

$$g(\bar{\boldsymbol{\beta}}) = \lambda_{\text{opt}}$$

- This can now be rewritten in the useful form

$$J = \sum_{k=1}^{N} w_k - g = \sum_{k=1}^{N} w_k - \lambda_{\text{opt}}$$

- Finally, the optimality condition can be written as

$$\lambda_{\text{opt}} = \sum_{k=1}^{N} w_k - J$$

- Note that *J* should be small for an optimal solution. This assumes that the measurement noise is reasonable small and Gaussian. The QUEST method then makes the elegant assumption that

$$\lambda_{\text{opt}} \approx \sum_{k=1}^{N} w_k$$

- This allows us to avoid the numerically intensive eigenvalue problem!

- However, we still need to find a solution for the eigenvector.

- The eigenvalues of [*K*] must satisfy the characteristic equation:

$$f(s) = \det([K] - s[I_{4\times 4}]) = 0$$

- The desire root can be solve using a classic Newton-Raphson iteration method:

$$\lambda_0 = \sum_{k=1}^{N} w_k$$

$$\lambda_1 = \lambda_0 - \frac{f(\lambda_0)}{f'(\lambda_0)}$$

$$\vdots$$

$$\lambda_{\max} = \lambda_i = \lambda_{i-1} - \frac{f(\lambda_{i-1})}{f'(\lambda_{i-1})}$$

- Let use introduce the classical Rodrigues parameter vector ***q***

$$\bar{\boldsymbol{q}} = \hat{\boldsymbol{e}} \tan\left(\frac{\Phi}{2}\right) = \frac{1}{\beta_0}\begin{bmatrix}\beta_1\\ \beta_2\\ \beta_3\end{bmatrix} = \frac{\boldsymbol{\epsilon}}{\beta_0}$$

- Note that

$$\frac{\bar{\boldsymbol{\beta}}}{\beta_0} = \begin{bmatrix}1\\ \bar{\boldsymbol{q}}\end{bmatrix}$$

- The eigenvector problem is now re-written as

$$[K]\frac{\bar{\boldsymbol{\beta}}}{\beta_0} = \lambda_{\text{opt}}\frac{\bar{\boldsymbol{\beta}}}{\beta_0}$$

$$\begin{bmatrix}\sigma & Z^T\\ Z & S - \sigma I_{3\times 3}\end{bmatrix}\begin{bmatrix}1\\ \bar{\boldsymbol{q}}\end{bmatrix} = \lambda_{\text{opt}}\begin{bmatrix}1\\ \bar{\boldsymbol{q}}\end{bmatrix}$$

$$\left([S] - \sigma[I_{3\times 3}]\right)\bar{\boldsymbol{q}} + [Z] = \lambda_{\text{opt}}\bar{\boldsymbol{q}}$$

- Finally, the classical Rodrigues parameter vector is found

$$\bar{\boldsymbol{q}} = \Big((\lambda_{\text{opt}} + \sigma)[I_{3\times3}] - [S]\Big)^{-1}[Z]$$

- Note that we still have to take an inverse of a 3x3 matrix here. However, this is numerically a very fast process.

- To solve for the corresponding 4-D Euler parameter vector, we use

$$\bar{\boldsymbol{\beta}} = \frac{1}{\sqrt{1 + \bar{\boldsymbol{q}}^T\bar{\boldsymbol{q}}}}\begin{bmatrix} 1 \\ \bar{\boldsymbol{q}} \end{bmatrix}$$

See Mathematica Solution of Example 3.16

Optimal Linear Attitude Estimator (OLAE)

Cayley Transform

$$[\bar{B}N] = ([I_{3\times3}] + [\tilde{\bar{\boldsymbol{q}}}])^{-1}([I_{3\times3}] - [\tilde{\bar{\boldsymbol{q}}}])$$

$$^{\mathcal{B}}\hat{\boldsymbol{v}}_i = [\bar{B}N]^{\mathcal{N}}\hat{\boldsymbol{v}}_i$$

$$([I_{3\times3}] + [\tilde{\bar{\boldsymbol{q}}}])\ ^{\mathcal{B}}\hat{\boldsymbol{v}}_i = ([I_{3\times3}] - [\tilde{\bar{\boldsymbol{q}}}])\ ^{\mathcal{N}}\hat{\boldsymbol{v}}_i$$

$$^{\mathcal{B}}\hat{\boldsymbol{v}}_i - {}^{\mathcal{N}}\hat{\boldsymbol{v}}_i = -[\tilde{\bar{\boldsymbol{q}}}](^{\mathcal{B}}\hat{\boldsymbol{v}}_i + {}^{\mathcal{N}}\hat{\boldsymbol{v}}_i)$$

Define:

$$\boldsymbol{s}_i = {}^{\mathcal{B}}\hat{\boldsymbol{v}}_i + {}^{\mathcal{N}}\hat{\boldsymbol{v}}_i$$

$$\boldsymbol{d}_i = {}^{\mathcal{B}}\hat{\boldsymbol{v}}_i - {}^{\mathcal{N}}\hat{\boldsymbol{v}}_i$$

$$\boldsymbol{d}_i = [\tilde{\boldsymbol{s}}_i]\bar{\boldsymbol{q}}$$

$$\boldsymbol{d} = \begin{bmatrix} \boldsymbol{d}_1 \\ \cdots \\ \boldsymbol{d}_N \end{bmatrix} \qquad [S] = \begin{bmatrix} \tilde{\boldsymbol{s}}_1 \\ \cdots \\ \tilde{\boldsymbol{s}}_N \end{bmatrix}$$

$$[W] = \begin{bmatrix} w_1 I_{3\times3} & 0_{3\times3} & \ddots \\ 0_{3\times3} & \ddots & 0_{3\times3} \\ \ddots & 0_{3\times3} & w_N I_{3\times3} \end{bmatrix}$$

$$\boxed{\bar{\boldsymbol{q}} = \left([S]^T[W][S]\right)^{-1}[S]^T[W]\boldsymbol{d}}$$

See Mathematica Solution of Example 3.17