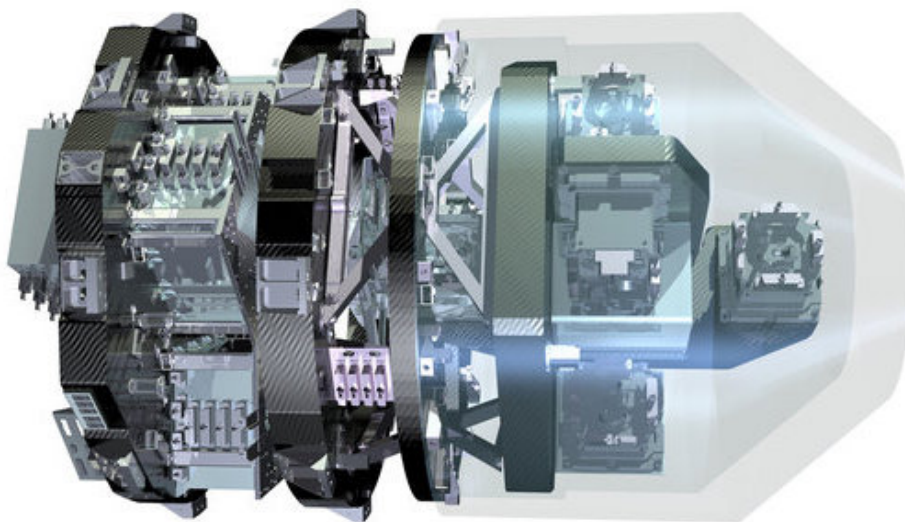


## TECHNICAL NOTE

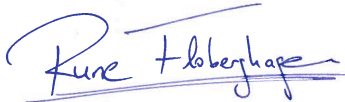
### GOCE Level 1B Gravity Gradient Processing Algorithms



Prepared by	Christian Siemes
Reference	RHEA for ESA - European Space Agency
Issue/Revision	ESA-EOPSM-GOCE-TN-3397
Date of Issue	1.0
Status	27/08/2018
	Approved



# APPROVAL

<b>Title</b> GOCE Level 1B Gravity Gradient Processing Algorithms	
<b>Issue Number</b> 1.0	<b>Revision Number</b> 0
<b>Author</b> Christian Siemes	<b>Date</b> 27/08/2018
<b>Approved by</b>	<b>Date of Approval</b>
<p><b>Rune Floberghagen (EOP-GM)</b> GOCE and Swarm Mission Manager</p> <p><b>Björn Frommknecht (EOP-GEP)</b> Payload Data GS Manager</p> <p><b>Roger Haagmans (EOP-SME)</b> GOCE Mission Scientist</p>	

# CHANGE LOG

Reason for change	Issue Nr.	Revision Number	Date
Document created	1.0	0	27/08/2018

# CHANGE RECORD

Issue Number 1.0	Revision Number 0		
Reason for change	Date	Pages	Paragraph(s)



# Contents

<b>1</b>	<b>Introduction</b>	<b>7</b>
1.1	Purpose of this document . . . . .	7
1.2	How to derive gravity gradients from accelerations . . . . .	7
1.3	Level 1b processing scheme . . . . .	8
<b>2</b>	<b>Basic processing elements</b>	<b>13</b>
2.1	Precision of epochs and time differences . . . . .	13
2.2	Interpolation . . . . .	14
2.3	Numerical integration . . . . .	14
2.4	Numerical differentiation . . . . .	15
<b>3</b>	<b>Convention for rotation matrices, quaternions, and angular rates</b>	<b>17</b>
<b>4</b>	<b>Star tracker data processing</b>	<b>21</b>
4.1	Star tracker data preprocessing . . . . .	21
4.2	Star tracker data combination . . . . .	21
<b>5</b>	<b>Adjusting the misalignment between star trackers and gradiometer</b>	<b>31</b>
<b>6</b>	<b>Accelerometer data processing</b>	<b>33</b>
6.1	Accelerometer data calibration . . . . .	33
6.2	Gross outlier detection and removal . . . . .	37
<b>7</b>	<b>Angular rate and acceleration reconstruction</b>	<b>43</b>
7.1	Calculation of star tracker angular rates . . . . .	43
7.2	Calculation of gradiometer angular rates . . . . .	44
7.3	Calculation of filters for angular rate reconstruction . . . . .	44
7.4	Application of filters for angular rate reconstruction . . . . .	46
<b>8</b>	<b>Attitude reconstruction</b>	<b>51</b>
8.1	Mathematical derivation of algorithm . . . . .	51
8.2	Covariance matrix . . . . .	54
8.3	Algorithm . . . . .	58
<b>9</b>	<b>Gravity gradient calculation</b>	<b>61</b>





## List of Algorithms

1	Numerical integration of a time series . . . . .	15
2	Numerical differentiation of a time series . . . . .	16
3	Converting rotation matrices to quaternions . . . . .	19
4	Making quaternions continuous . . . . .	23
5	Resampling of star tracker quaternions to epochs of gradiometer (part 1) . . . . .	25
5	Resampling of star tracker quaternions to epochs of gradiometer (part 2) . . . . .	26
6	Star tracker combination (part 1) . . . . .	28
6	Star tracker combination (part 2) . . . . .	29
6	Star tracker combination (part 3, optional) . . . . .	30
7	Adjustment of the misalignment between star trackers and gradiometer . . . . .	31
8	Gradiometer calibration (satellite shaking) . . . . .	37
9	Gradiometer calibration (science mode) . . . . .	38
10	Gross outlier removal (part 1) . . . . .	39
10	Gross outlier removal (part 2) . . . . .	40
11	Calculation of star tracker angular rates . . . . .	45
12	Calculation of gradiometer angular rates . . . . .	45
13	Calculation of filters for angular rate reconstruction . . . . .	47
14	Angular rate reconstruction (part 1) . . . . .	48
14	Angular rate reconstruction (part 2) . . . . .	49
15	Attitude reconstruction algorithm (part 1) . . . . .	59
15	Attitude reconstruction algorithm (part 2) . . . . .	60
16	Gravity gradient calculation algorithm . . . . .	61





# 1 Introduction

The GOCE satellites carries a gravity gradiometer, consisting of six accelerometers, and three star trackers as part of the payload. The gravity gradients are calculated from the measurements of these instruments. In the remainder of this section, we provide a high-level description of the processing, whereas more details are provided in the following sections.

## 1.1 Purpose of this document

The purpose of this document is to describe the processing scheme and algorithms in such detail, that it is possible to implement the GOCE Level 1b processing and arrive at the same results within numerical precision. It forms the fundamental basis for the GOCE Level 1b reprocessing performed in the year 2018, in the sense that the reprocessed gravity gradients and attitude quaternions were calculated following the instruction in this document.

## 1.2 How to derive gravity gradients from accelerations

A perfect accelerometer onboard a satellite measures the acceleration

$$\mathbf{a}_i = -(\mathbf{V} - \mathbf{\Omega}^2 - \dot{\mathbf{\Omega}})\mathbf{r}_i + \mathbf{d}, \quad (1)$$

where  $i$  is the identifier of the accelerometer,  $\mathbf{r}_i$  is the vector from the satellite's centre of mass to the proof mass centre of the  $i$ -th accelerometer,  $\mathbf{V}$  contains the gravity gradients,  $\mathbf{\Omega}^2\mathbf{r}_i$  are centrifugal accelerations,  $\dot{\mathbf{\Omega}}\mathbf{r}_i$  are Euler accelerations, and  $\mathbf{d}$  are non-gravitational accelerations. The matrices  $\mathbf{V}$ ,  $\dot{\mathbf{\Omega}}$  and  $\mathbf{\Omega}^2$  are defined as

$$\mathbf{V} = \begin{bmatrix} V_{xx} & V_{xy} & V_{xz} \\ V_{xy} & V_{yy} & V_{yz} \\ V_{xz} & V_{yz} & V_{zz} \end{bmatrix}, \quad (2)$$

$$\dot{\mathbf{\Omega}} = \begin{bmatrix} 0 & -\dot{\omega}_z & \dot{\omega}_y \\ \dot{\omega}_z & 0 & -\dot{\omega}_x \\ -\dot{\omega}_y & \dot{\omega}_x & 0 \end{bmatrix} \quad (3)$$

and

$$\mathbf{\Omega}^2 = \begin{bmatrix} -\omega_y^2 - \omega_z^2 & \omega_x\omega_y & \omega_x\omega_z \\ \omega_x\omega_y & -\omega_x^2 - \omega_z^2 & \omega_y\omega_z \\ \omega_x\omega_z & \omega_y\omega_z & -\omega_x^2 - \omega_y^2 \end{bmatrix}, \quad (4)$$



respectively. In order to extract the gravity gradients from the accelerations  $\mathbf{a}_i$ , we build the following sums and differences. First, we calculate common mode accelerations

$$\mathbf{a}_{c,ij} = \frac{1}{2}(\mathbf{a}_i + \mathbf{a}_j) = \mathbf{d} \quad (5)$$

and differential mode accelerations

$$\mathbf{a}_{d,ij} = \frac{1}{2}(\mathbf{a}_i - \mathbf{a}_j) = -(\mathbf{V} - \mathbf{\Omega}^2 - \dot{\mathbf{\Omega}})\mathbf{r}_{dij} \quad (6)$$

where the differential accelerometer positions  $\mathbf{r}_{dij}$  are defined in the same way as the differential mode accelerations, i.e.

$$\mathbf{r}_{d,ij} = \frac{1}{2}(\mathbf{r}_i - \mathbf{r}_j). \quad (7)$$

The non-gravitational acceleration  $\mathbf{d}$  is separated in this way. Next, we define the matrices

$$\mathbf{A}_d = \begin{bmatrix} \mathbf{a}_{d14} & \mathbf{a}_{d25} & \mathbf{a}_{d36} \end{bmatrix} \quad (8)$$

and

$$\mathbf{R}_d = \begin{bmatrix} \mathbf{r}_{d14} & \mathbf{r}_{d25} & \mathbf{r}_{d36} \end{bmatrix} = \begin{bmatrix} L_x & 0 & 0 \\ 0 & L_y & 0 \\ 0 & 0 & L_z \end{bmatrix}, \quad (9)$$

where  $L_x$ ,  $L_y$  and  $L_z$  are the length of the gradiometer arms. We use the matrices to calculate

$$\mathbf{A}_d \mathbf{R}_d^{-1} + (\mathbf{A}_d \mathbf{R}_d^{-1})^T = -2(\mathbf{V} - \mathbf{\Omega}^2) \quad (10)$$

and

$$\mathbf{A}_d \mathbf{R}_d^{-1} - (\mathbf{A}_d \mathbf{R}_d^{-1})^T = 2\dot{\mathbf{\Omega}}. \quad (11)$$

This step separated the Euler acceleration from the gravity gradient. The last task is to determine the centrifugal acceleration in order to find the gravity gradient. For this purpose we combine the angular accelerations  $\dot{\mathbf{\Omega}}$  measured by the gradiometer with the star tracker attitude in order to find the angular rates  $\mathbf{\Omega}$ . Once the angular acceleration is calculated, we can also calculate the term  $\mathbf{\Omega}^2$ .

### 1.3 Level 1b processing scheme

The GOCE Level 1b data reprocessing is illustrated in Fig. 1, where each red box represents an algorithm that is described in detail in the remainder of this document. Here, we provide only a brief description of the algorithm including its significance in the larger processing scheme and key changes in comparison to the original processing.





**EGG calibration (shaking mode)** A so-called satellite shaking procedure was executed approximately every two months and lasted 24 hours each time. The data collected during a satellite shaking was used to determine the inverse calibration matrices that were used in the original processing. In the reprocessing, these inverse calibration matrices are also applied in order to arrive at pre-calibrated acceleration data. This step is important for the EGG calibration algorithm, which would fail when nominal (uncalibrated) acceleration data were used as input. The main reason is the that the gradiometer angular rates calculated from nominal acceleration data are affected by large errors.

**EGG outlier detection and removal** The acceleration data is occasionally affected by gross outliers, which are potentially caused by micro-vibrations onboard the GOCE satellite. These outliers need to be identified and removed from the acceleration data in order to prevent them from entering the calculation of the gradiometer angular rates. The latter includes an integration of the acceleration data that would transform the gross outlier into a step function. The step function would then be high-pass filtered in the angular rate reconstruction, which "smears out" the effect of the gross outlier, making it difficult to remove it from the gravity gradient data.

**EGG calibration (science mode)** A detailed comparison to gravity gradients calculated from a GRACE gravity field model revealed that the measured GOCE gravity gradients are affected by small perturbations caused by imperfect inverse calibration matrices determined from the satellite shakings. In addition, it was found that an unmodeled quadratic factor was causing perturbations in the gravity gradient  $V_{yy}$  predominantly in the regions around the geomagnetic poles. The purpose of the gradiometer calibration in the so-called science mode, a flight operation mode that ensures a "quiet" environment for the gradiometer, is to correct for the imperfections in the inverse calibration matrices and the quadratic factors.

**STR preprocessing** The star tracker attitude and CCD temperature data are available at different sampling rates and epochs than the gradiometer data. The star tracker preprocessing resamples the star tracker attitude and CCD temperature data to the epochs of the gradiometer data.

**STR combination** The star tracker combination combines the attitude data from all available star trackers into a single attitude quaternion. The addition to the original star tracker combination is the correction of relative, temperature-dependent star tracker attitude biases.

**STR misalignment correction** As in the original L1b processing, the small misalignments between the star tracker assembly and the gradiometer are corrected in this processing step. The misalignments used for correction are however different ones, which are consistent with the inverse calibration matrices used in the science mode gradiometer calibration.

**Angular rate and acceleration reconstruction** The algorithm for the angular rate and acceleration reconstruction remains the same to a large extend, where the main difference



is the use of different relative weights for the gradiometer and star tracker angular rates. A small addition to the algorithm avoids the edge effects due to the filtering, which is part of the angular rate reconstruction.

**Attitude reconstruction** The attitude reconstruction is a completely new algorithm. In the original algorithm, a single star tracker quaternion was integrated over more than one orbit using the reconstructed angular rates. The resulting time series of integrated star tracker quaternions was merged with the attitude quaternions from the star tracker combination by applying a high-pass filter to the first and a complementary low-pass filter to the latter. The weak point of this approach is the propagation of the attitude error of the star tracker quaternion used to initialise the integration. Even in case of error-free angular rates, the error in the integrated quaternion grows proportional to the distance from the initial quaternion. This error propagation is taken into account in the new algorithm, where the reconstructed attitude quaternion is estimated by fitting the reconstructed angular rates to differences of the quaternions from the star tracker combination.

**Gravity gradient calculation** The calculation of the gravity gradients remains the same.

Table 1: GOCE L1b algorithms referenced within processing scheme

Name of algorithm (cf. Fig. 1)	Algorithm listing
EKG calibration (shaking mode)	Algorithm 8
EKG outlier detection and removal	Algorithm 10
EKG calibration (science mode)	Algorithm 9
STR preprocessing	Algorithm 5
STR combination	Algorithm 6
STR misalignment correction	Algorithm 7
Angular rate and acceleration reconstruction	Algorithms 13, 14
Attitude reconstruction	Algorithm 15
Gravity gradient calculation	Algorithm 16

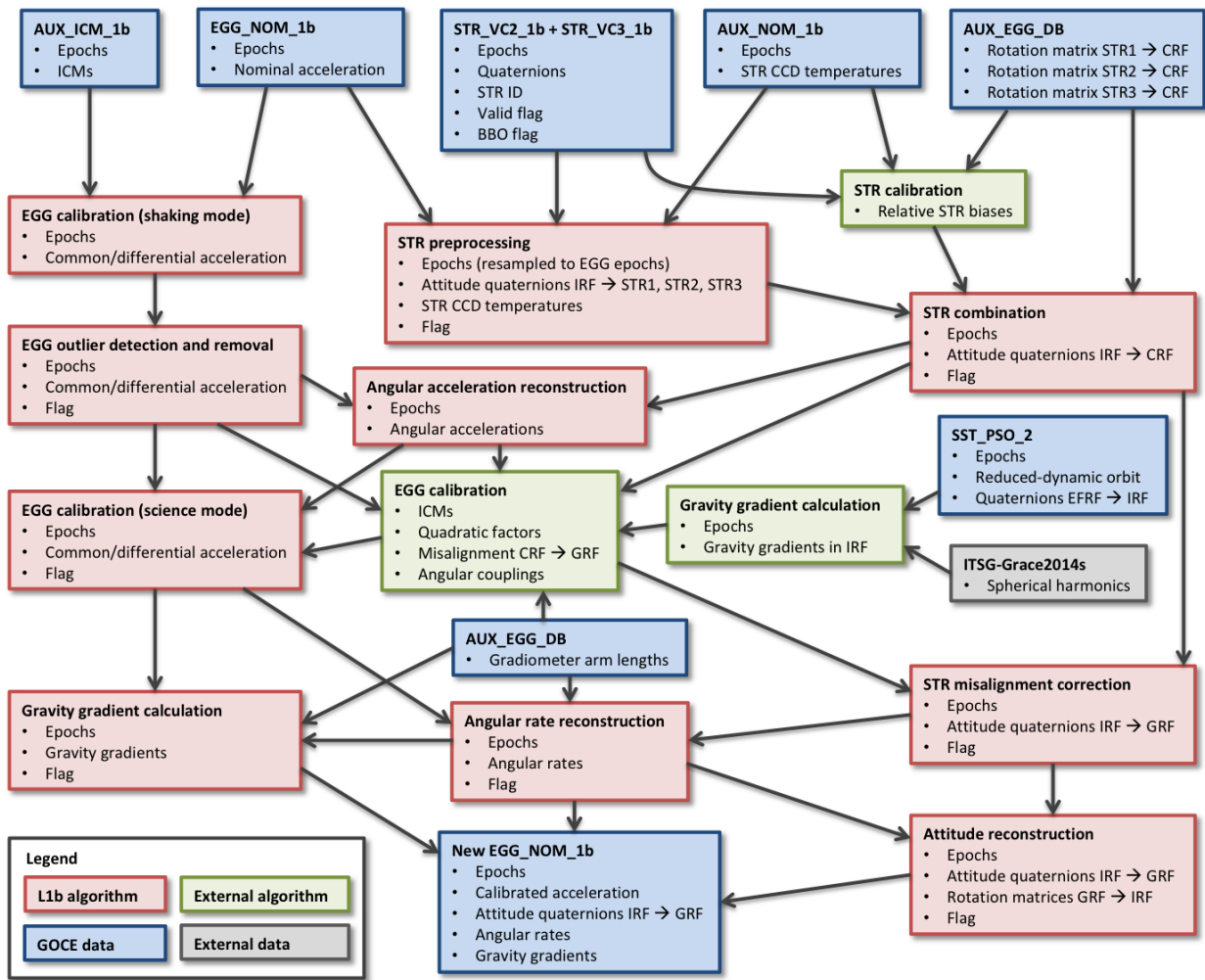


Figure 1: Flowchart for L1b data reprocessing and calibration.



## 2 Basic processing elements

### 2.1 Precision of epochs and time differences

The algorithms described in this document include often the calculation of time differences between epochs. In order to reproduce the GOCE gravity gradients with a precision better than 1 mE, the time differences require a precision of at least nano-second level. In order to achieve that precision for time differences, the epochs need to be stored with at least the same precision.

For GOCE mission data, the epochs are specified typically as GPS seconds, which are during mission lifetime in the order of  $10^{10}$  seconds. Thus, representing the epochs with double precision is insufficient because then the precision of the epochs would be  $10^{10} \times 2^{-52} \approx 10^{-6}$  seconds, i.e. at micro-second level. There are many ways of increasing the precision of the epochs. One way is using two double precision variables for representing the epochs  $t_n$ , where the first double precision variable represents the integer part  $t_n^{int}$  of the GPS second and the other double precision variable represents the sub-second part  $t_n^{sub}$  of the GPS second. The integer and sub-second part of the GPS second are obtained by

$$t_n^{int} = \text{floor}(t_n) \quad (12)$$

and

$$t_n^{sub} = t_n - \text{floor}(t_n), \quad (13)$$

respectively, such that

$$t_n = t_n^{int} + t_n^{sub}. \quad (14)$$

The time difference between two arbitrary epochs  $t_k$  and  $t_n$  can then be calculated with the required precision by

$$t_k - t_n = \text{round}(t_k^{int} - t_n^{int}) + t_k^{sub} - t_n^{sub}. \quad (15)$$

When an already existing software routine is used to perform a calculation, it may not be possible to apply Eq. (15) strictly. For example, Matlab's `interp1` function accepts only one double variable for the epochs  $t_n$ . However, in many cases the software routine does not perform calculations on the epochs  $t_n$  directly, but only on time differences between epochs. In such cases, the time differences  $\Delta t_n$  between all epochs and the first epoch,

$$\Delta t_n = \text{round}(t_n^{int} - t_1^{int}) + t_n^{sub} - t_1^{sub}, \quad (16)$$

can replace the epochs  $t_n$  as input variable to the existing software routine. If the input data to the routine does not span more than 90 days, the precision of the time differences  $\Delta t_n$  is  $90 \times 86400 \times 2^{-52} \approx 10^{-9}$  seconds, i.e. the precision of  $\Delta t_n$  is at the required nano-second level.

In all algorithms described in this document, either Eq. (15) or Eq. (16) shall be used to calculate time differences, noting that we will not explicitly refer to these equations.



## 2.2 Interpolation

The method used for all interpolations is cubic spline interpolation with the following conditions:

- The cubic splines interpolate the data points.
- The first and second derivative is a continuous function.
- The third derivative is continuous in the second data point as well as the second last data point ("not-a-knot" condition).

This is known as cubic spline interpolation with "not-a-knot" conditions, which is Matlab's default method of spline interpolation of the `interp1` function. In this document, we denote this interpolation by

$$\mathbf{x}^{interp} = \text{interpolate}(\mathbf{t}, \mathbf{x}, \mathbf{t}^{interp}) \quad (17)$$

where vector  $\mathbf{t}$  contains the original epochs, vector  $\mathbf{x}$  contains the original data points, vector  $\mathbf{t}^{int}$  contains the epochs to which we interpolate, and vector  $\mathbf{x}^{int}$  contains the interpolated data points.

## 2.3 Numerical integration

For numerical integration of a time series, whose values  $\mathbf{x} = [x_1 \dots x_N]$  are given at epochs  $\mathbf{t} = [t_1 \dots t_N]$ , our approach is interpolating  $\mathbf{x}$  using cubic spline interpolation with "not-a-knot" conditions and then integrating the splines. Since the software available to us does not support analytical integration of the splines, we first upsample the time series and then apply trapezoidal integration on the upsampled time series. We denote the epochs and values of the upsampled time series by  $\mathbf{t}^{up}$  and  $\mathbf{x}^{up}$ , respectively. We specify the increase of the sampling rate by an integer factor  $K$ , such that between each two epochs  $t_n$  and  $t_{n+1}$  we insert  $K - 1$  new epochs that are equally spaced between  $t_n$  and  $t_{n+1}$ . A typical value is  $K = 20$  for processing described in this document. Then, we interpolate the time series  $\mathbf{x}$  to the epochs  $\mathbf{t}^{up}$  and reduce the mean from the resulting upsampled time series  $\mathbf{x}^{up}$  in order to keep the accumulation of rounding errors low in the following step, which is the trapezoidal integration of  $\mathbf{x}^{up}$ . Finally, we decimate the integrated time series  $\mathbf{x}^{up,int}$  to the original epochs to obtain the integrated time series  $\mathbf{x}^{int}$ . All of these steps are detailed in Algorithm 1 for numerical integration of a time series. For convenience, we denote the numerical integration by

$$\mathbf{x}^{int} = \text{integrate}(\mathbf{t}, \mathbf{x}, K). \quad (18)$$

Inputs	Symbol	Unit
Epochs	$\mathbf{t}$	Time unit, e.g. GPS seconds
Values	$\mathbf{x}$	Any unit
Upsampling factor	$K$	unitless
Outputs	Symbol	Unit
Integrated time series	$\mathbf{x}^{int}$	Any unit multiplied by time unit

Table 2: List of inputs and outputs of numerical integration algorithm

**Algorithm 1** Numerical integration of a time series

---

```

1:  $\mathbf{x}^{int} = \text{zeros}(N, 1)$ 
2:  $\mathbf{t}^{up} = \text{zeros}((N - 1)K + 1, 1)$ 
3:  $\mathbf{x}^{int,up} = \text{zeros}((N - 1)K + 1, 1)$ 
4: for  $n \leftarrow 1, N - 1$  do
5:   for  $k \leftarrow 0, K - 1$  do
6:      $t_{1+(n-1)K+k}^{up} = t_n + \frac{k}{K}(t_{n+1} - t_n)$ 
7:   end for
8: end for
9:  $t_{(N-1)K+1}^{up} = t_N$ 
10:  $\mathbf{x}^{up} = \text{interpolate}(\mathbf{t}, \mathbf{x}, \mathbf{t}^{up})$ 
11:  $\mathbf{x}^{up} = \mathbf{x}^{up} - \text{mean}(\mathbf{x}^{up})$ 
12: for  $n \leftarrow 2, (N - 1)K + 1$  do
13:    $x_n^{int,up} = x_{n-1}^{int,up} + (x_n^{up} + x_{n-1}^{up})(t_n^{up} - t_{n-1}^{up})/2$ 
14: end for
15: for  $n \leftarrow 1, N$  do
16:    $x_n^{int} = x_{1+(n-1)K}^{int,up}$ 
17: end for

```

---

## 2.4 Numerical differentiation

For calculating the first time derivative of a time series, whose values  $\mathbf{x} = [x_1 \dots x_N]$  are given at epochs  $\mathbf{t} = [t_1 \dots t_N]$ , we interpolate  $\mathbf{x}$  to the epochs  $\mathbf{t} - \Delta t$  and  $\mathbf{t} + \Delta t$  using cubic spline interpolation with "not-a-knot" conditions, where  $\Delta t$  is small in comparison to the time difference  $t_{n+1} - t_n$ . For processing described in this document, typical values are  $t_{n+1} - t_n = 1$  second and  $\Delta t = 1$  millisecond. The resulting interpolated values are denoted by  $\mathbf{x}^{-\Delta t}$  and  $\mathbf{x}^{+\Delta t}$ , respectively. When using Matlab's `interp1` function for the calculation of  $\mathbf{x}^{-\Delta t}$  and  $\mathbf{x}^{+\Delta t}$ , extrapolation has to be switched on. Then, we obtain the first time derivative  $\dot{\mathbf{x}}$  by calculating

$$\dot{\mathbf{x}} = \frac{\mathbf{x}^{+\Delta t} - \mathbf{x}^{-\Delta t}}{2\Delta t}. \quad (19)$$



The calculation is summarised in Algorithm 2 for numerical differentiation of a time series. For convenience, we denote the numerical differentiation by

$$\dot{\mathbf{x}} = \text{differentiate}(\mathbf{t}, \mathbf{x}, \Delta t). \quad (20)$$

Inputs	Symbol	Unit
Epochs	$\mathbf{t}$	Time unit, e.g. GPS seconds
Values	$\mathbf{x}$	Any unit
Upsampling factor	$\Delta t$	unitless
Outputs	Symbol	Unit
Integrated time series	$\dot{\mathbf{x}}$	Any unit divided by time unit

Table 3: List of inputs and outputs of numerical differentiation algorithm

---

**Algorithm 2** Numerical differentiation of a time series

---

- 1:  $\mathbf{x}^{-\Delta t} = \text{interpolate}(\mathbf{t}, \mathbf{x}, \mathbf{t} - \Delta t)$
  - 2:  $\mathbf{x}^{+\Delta t} = \text{interpolate}(\mathbf{t}, \mathbf{x}, \mathbf{t} + \Delta t)$
  - 3:  $\dot{\mathbf{x}} = \frac{\mathbf{x}^{+\Delta t} - \mathbf{x}^{-\Delta t}}{2\Delta t}$
-



### 3 Convention for rotation matrices, quaternions, and angular rates

We use the conventions for quaternions, rotations and angular rates as described in Groves [2013], which we repeat here for convenience. A rotation is defined by

$$\mathbf{x}^B = \mathbf{R}_A^B \mathbf{x}^A \quad (21)$$

where  $\mathbf{x}^A$  is a vector in the  $A$ -frame,  $\mathbf{x}^B$  is a vector in the  $B$ -frame, and  $\mathbf{R}_A^B$  is the rotation matrix that rotates from the  $A$ -frame to the  $B$ -frame. One way to represent a rotation matrix by a sequence of elementary rotations is

$$\mathbf{R}_A^B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi_A^B & \sin \phi_A^B \\ 0 & -\sin \phi_A^B & \cos \phi_A^B \end{bmatrix} \begin{bmatrix} \cos \theta_A^B & 0 & -\sin \theta_A^B \\ 0 & 1 & 0 \\ \sin \theta_A^B & 0 & \cos \theta_A^B \end{bmatrix} \begin{bmatrix} \cos \psi_A^B & \sin \psi_A^B & 0 \\ -\sin \psi_A^B & \cos \psi_A^B & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (22)$$

The frame transformation for the gravity gradient tensor  $\mathbf{V}$  reads

$$\mathbf{V}^B = \mathbf{R}_A^B \mathbf{V}^A (\mathbf{R}_A^B)^T = \mathbf{R}_A^B \mathbf{V}^A \mathbf{R}_B^A. \quad (23)$$

In case the angles of rotation  $\phi_A^B$ ,  $\theta_A^B$  and  $\psi_A^B$  are small, we can approximate

$$\mathbf{R}_A^B = \begin{bmatrix} 1 & \psi_A^B & -\theta_A^B \\ -\psi_A^B & 1 & \phi_A^B \\ \theta_A^B & -\phi_A^B & 1 \end{bmatrix}. \quad (24)$$

The angular rate vector is denoted by  $\boldsymbol{\omega}_{B,A}^C$  and describes the rate of rotation of the  $A$ -frame axes with respect to the  $B$ -frame axes, resolved about the  $C$ -frame axes. The skew symmetric matrix

$$\boldsymbol{\Omega}_{B,A}^C = \begin{bmatrix} 0 & -\omega_{B,A,z}^C & \omega_{B,A,y}^C \\ \omega_{B,A,z}^C & 0 & -\omega_{B,A,x}^C \\ -\omega_{B,A,y}^C & \omega_{B,A,x}^C & 0 \end{bmatrix} \quad (25)$$

is also commonly used for the angular rate vector. The first time derivative of the rotation matrix is related to the angular rates by

$$\dot{\mathbf{R}}_A^B = -\boldsymbol{\Omega}_{A,B}^B \mathbf{R}_A^B \quad (26)$$

where we assume that  $B$ -frame axes are rotating with respect to the stationary  $A$ -frame axes. In the context of the GOCE mission, the GRF is thus equivalent to the  $B$ -frame and the IRF is equivalent to the  $A$ -frame.

In some situation it is more practical to work with quaternions instead of rotation matrices. A quaternion that describes the same rotation than  $\mathbf{R}_A^B$  is defined as

$$q_A^B = \begin{bmatrix} q_{A,0}^B & q_{A,1}^B & q_{A,2}^B & q_{A,3}^B \end{bmatrix}^T \quad (27)$$

where  $q_{A,0}^B$  is the real element of the quaternion and  $q_{A,1}^B$ ,  $q_{A,2}^B$  and  $q_{A,3}^B$  are imaginary elements of the quaternions. In the context of the GOCE mission,  $q_{A,0}^B$  is labelled  $q_{A,4}^B$ . The rotation matrix and the quaternion are related by

$$\mathbf{R}_A^B = \begin{bmatrix} q_{A,0}^{B^2} + q_{A,1}^{B^2} - q_{A,2}^{B^2} - q_{A,3}^{B^2} & 2(q_{A,1}^B q_{A,2}^B + q_{A,3}^B q_{A,0}^B) & 2(q_{A,1}^B q_{A,3}^B - q_{A,2}^B q_{A,0}^B) \\ 2(q_{A,1}^B q_{A,2}^B - q_{A,3}^B q_{A,0}^B) & q_{A,0}^{B^2} - q_{A,1}^{B^2} + q_{A,2}^{B^2} - q_{A,3}^{B^2} & 2(q_{A,2}^B q_{A,3}^B + q_{A,1}^B q_{A,0}^B) \\ 2(q_{A,1}^B q_{A,3}^B + q_{A,2}^B q_{A,0}^B) & 2(q_{A,2}^B q_{A,3}^B - q_{A,1}^B q_{A,0}^B) & q_{A,0}^{B^2} - q_{A,1}^{B^2} - q_{A,2}^{B^2} + q_{A,3}^{B^2} \end{bmatrix}. \quad (28)$$

The sequence of rotations from the  $A$ -frame to the  $C$ -frame via the  $B$ -frame can be performed in terms of rotation matrix multiplications

$$\mathbf{R}_A^C = \mathbf{R}_B^C \mathbf{R}_A^B \quad (29)$$

or equivalently in terms of quaternion multiplications

$$q_A^C = q_A^B q_B^C, \quad (30)$$

noting that the sequence of quaternion multiplications is reversed compared to that of rotation matrix multiplications.

For small rotation angles, we can approximate the quaternion  $q_A^B$  by

$$q_A^B = \begin{bmatrix} 1 & \phi_A^B/2 & \theta_A^B/2 & \psi_A^B/2 \end{bmatrix}^T. \quad (31)$$

For small time intervals  $\Delta t$ , we can relate the small rotation angles to the angular rates by

$$q_A^B(t + \Delta t) = q_A^B(t) q_B^{B(t+\Delta t)} \quad (32)$$

where

$$q_B^{B(t+\Delta t)} = \begin{bmatrix} 1 \\ \phi_{B(t)}^{B(t+\Delta t)}/2 \\ \theta_{B(t)}^{B(t+\Delta t)}/2 \\ \psi_{B(t)}^{B(t+\Delta t)}/2 \end{bmatrix} = \begin{bmatrix} 1 \\ \int_t^{t+\Delta t} \omega_{A,B,x}^B dt/2 \\ \int_t^{t+\Delta t} \omega_{A,B,y}^B dt/2 \\ \int_t^{t+\Delta t} \omega_{A,B,z}^B dt/2 \end{bmatrix}. \quad (33)$$

Equation (26) expressed in terms of quaternions reads

$$\dot{q}_A^B = q_A^B \mathbf{W}_{A,B}^B \quad (34)$$

where the product  $\mathbf{W}_{A,B}^B q_A^B$  is a quaternion multiplication and

$$\mathbf{W}_{A,B}^B = \begin{bmatrix} 0 & \omega_{A,B,x}^B/2 & \omega_{A,B,y}^B/2 & \omega_{A,B,z}^B/2 \end{bmatrix}^T \quad (35)$$

is a vector that contains the angular rates. Note that the different sign in Eq. (34) with respect to Eq. (26) results from the different signs in Eq. (24) and Eq. (25).

---

**Algorithm 3** Converting rotation matrices to quaternions
 

---

```

1:  $t = R_{11} + R_{22} + R_{33}$ 
2: if  $t > 0$  then
3:    $r = \sqrt{1 + t}$ 
4:    $s = 0.5/r$ 
5:    $q_0 = 0.5r$ 
6:    $q_1 = (R_{32} - R_{23})s$ 
7:    $q_2 = (R_{13} - R_{31})s$ 
8:    $q_3 = (R_{21} - R_{12})s$ 
9: else if  $R_{11} > R_{22}$  and  $R_{11} > R_{33}$  then
10:   $r = \sqrt{1 + R_{11} - R_{22} - R_{33}}$ 
11:   $s = 0.5/r$ 
12:   $q_0 = (R_{32} - R_{23})s$ 
13:   $q_1 = 0.5r$ 
14:   $q_2 = (R_{12} + R_{21})s$ 
15:   $q_3 = (R_{31} + R_{13})s$ 
16: else if  $R_{22} > R_{11}$  and  $R_{22} > R_{33}$  then
17:   $r = \sqrt{1 + R_{22} - R_{11} - R_{33}}$ 
18:   $s = 0.5/r$ 
19:   $q_0 = (R_{13} - R_{31})s$ 
20:   $q_1 = (R_{21} + R_{12})s$ 
21:   $q_2 = 0.5r$ 
22:   $q_3 = (R_{32} + R_{23})s$ 
23: else
24:   $r = \sqrt{1 + R_{33} - R_{11} - R_{22}}$ 
25:   $s = 0.5/r$ 
26:   $q_0 = (R_{21} - R_{12})s$ 
27:   $q_1 = (R_{31} + R_{13})s$ 
28:   $q_2 = (R_{32} + R_{23})s$ 
29:   $q_3 = 0.5r$ 
30: end if

```

▷  $R_{33} > R_{11}$  and  $R_{33} > R_{22}$

---





## 4 Star tracker data processing

### 4.1 Star tracker data preprocessing

On board the GOCE satellite are three star trackers, each providing its orientation with respect to the international celestial reference frame (IRF). The orientation is provided in form of an attitude quaternion  $q_{IRF}^{SRF_i}$ , where  $i \in \{1, 2, 3\}$  indicates the star tracker. The star trackers also provide various flags that give information on the tracking status. In the first step of the star tracker data preprocessing, we use the validity flag  $f_{VAL_i}$  and the big-bright-object flag  $f_{BBO_i}$ , which indicate whether the star tracker is providing a valid attitude and whether a big and bright object is within the field-of-view, respectively. We discard all attitude quaternions that are flagged invalid, i.e.  $f_{VAL_i} = 0$  and for which a big-bright-object is detected, i.e.  $f_{BBO_i} = 1$ .

Since the measurements of the star trackers are not synchronised with the gradiometer measurements, we resample all star tracker data to the measurement epochs of the gradiometer. The first step is loading the star tracker epochs, quaternions and flags from the STR\_VC3.1B and STR\_VC3\_1B files and star tracker CCD temperatures from the AUX\_NOM\_1B files, followed by sorting all loaded data into individual variables for each star tracker. Then, we amend for each star tracker all quaternion sign flips between subsequent epochs using Algorithm 4, noting that  $q_{IRF}^{SRF_i}$  and  $-q_{IRF}^{SRF_i}$  describe the same attitude. For the resampling we select all quaternions in a time window  $[t_{G,n} - \Delta t_q, t_{G,n} + \Delta t_q]$  centred around the gradiometer epochs  $t_{G,n}$  and approximate them with a quadratic function, provided that we have at least three quaternions within the time window and at least one quaternion on each side of  $t_{G,n}$ . We use the same approach to resample the star tracker CCD temperatures to the gradiometer epochs, with the differences that we use a larger time window  $[t_{G,n} - \Delta t_T, t_{G,n} + \Delta t_T]$  and that we approximate the temperatures by their mean value within the time window. These processing steps constitute the star tracker preprocessing, which is detailed in Algorithm 5. Typical values for the time windows are  $\Delta t_q = 1.75$  seconds, i.e. up to 7 star tracker epochs due to the sampling rate of 2 Hz for quaternions, and  $\Delta t_T = 300$  seconds, i.e. up to 38 epochs due to the sampling rate of 1/16 Hz for temperatures. Further, it is noted that the resolution for the temperatures is limited to roughly  $0.5^\circ C$ .

### 4.2 Star tracker data combination

We use the approach of Romans [2003] for the combination of the attitude quaternions. It is based on a least squares adjustment of the star tracker quaternions, in which the pointing of the star tracker bore sight is assumed to be 10 times more accurate compared to the rotation around the bore sight. It requires knowledge of the orientation of the star trackers in the common reference frame (CRF), which is aligned with the satellite's body axes. That orientation is available in form the rotation matrices  $\mathbf{R}_{SRF_1}^{CRF}$ ,  $\mathbf{R}_{SRF_2}^{CRF}$  and  $\mathbf{R}_{SRF_3}^{CRF}$  defined in Eqs. (36–38). Our



addition to the approach of Romans [2003] is the correction of relative biases between the star trackers, which are modelled as linear functions of the temperature as defined in Eqs. (39–44). Algorithm 6 shows all processing steps of the star tracker combination in detail.

$$\mathbf{R}_{SRF_1}^{CRF} = \begin{bmatrix} 0.999991953964000 & -0.003855453067860 & 0.001107921250810 \\ -0.002875276132160 & -0.496285685373000 & 0.868154508875000 \\ -0.002797283507320 & -0.868150709252000 & -0.496292777733000 \end{bmatrix} \quad (36)$$

$$\mathbf{R}_{SRF_2}^{CRF} = \begin{bmatrix} 0.999868439135000 & 0.015726793513000 & -0.003971446564830 \\ 0.016149312081100 & -0.942268716879000 & 0.334468032720000 \\ 0.001517939828470 & -0.334488165946000 & -0.942398728087000 \end{bmatrix} \quad (37)$$

$$\mathbf{R}_{SRF_3}^{CRF} = \begin{bmatrix} 0.011846242780200 & -0.769183928773000 & 0.638917639645000 \\ -0.491411293086000 & 0.551999304112000 & 0.673655482637000 \\ -0.870847063243000 & -0.321951629871000 & -0.371446551289000 \end{bmatrix} \quad (38)$$

$$\mathbf{b}_{S_1}^{CRF}(T_{S_1}) = \mathbf{b}_{C_1}^{CRF} + T_{S_1} \mathbf{b}_{T_1}^{CRF} \quad (39)$$

$$\mathbf{b}_{C_1}^{CRF} = 10^{-3} \begin{bmatrix} 0.116219900793661 \\ -0.134723547186391 \\ -0.029472128350279 \end{bmatrix}, \quad \mathbf{b}_{T_1}^{CRF} = 10^{-5} \begin{bmatrix} 0.278591682091328 \\ -0.118889821498250 \\ -0.140330884420176 \end{bmatrix} \quad (40)$$

$$\mathbf{b}_{S_2}^{CRF}(T_{S_2}) = \mathbf{b}_{C_2}^{CRF} + T_{S_2} \mathbf{b}_{T_2}^{CRF} \quad (41)$$

$$\mathbf{b}_{C_2}^{CRF} = 10^{-3} \begin{bmatrix} 0.087909010253279 \\ -0.223645453432216 \\ -0.007718724727271 \end{bmatrix}, \quad \mathbf{b}_{T_2}^{CRF} = 10^{-5} \begin{bmatrix} 0.046609082258701 \\ 0.226425836947881 \\ -0.096374884840557 \end{bmatrix} \quad (42)$$

$$\mathbf{b}_{S_3}^{CRF}(T_{S_3}) = \mathbf{b}_{C_3}^{CRF} + T_{S_3} \mathbf{b}_{T_3}^{CRF} \quad (43)$$

$$\mathbf{b}_{C_3}^{CRF} = 10^{-3} \begin{bmatrix} 0.111289309287413 \\ -0.147455472014728 \\ 0.021704225770305 \end{bmatrix}, \quad \mathbf{b}_{T_3}^{CRF} = 10^{-5} \begin{bmatrix} 0.053953847437714 \\ -0.064274246885287 \\ 0.379499278972736 \end{bmatrix} \quad (44)$$



Inputs	Symbol	Unit	Contents
Flags	$f$	unitless	0 = invalid, 1 = valid
Quaternions	$q$	unitless	Quaternions with sign flips
Outputs	Symbol	Unit	Contents
Quaternions	$q$	unitless	Quaternions without sign flips

Table 4: List of inputs and outputs of algorithm for making quaternions continuous

---

**Algorithm 4** Making quaternions continuous

---

```

1: for  $n \leftarrow 2, N$  do
2:   if  $f_n == 0$  then                                     ▷ No valid attitude available
3:      $q_n = q_{n-1}$ 
4:   else if  $q_n^T q_{n-1} < 0$  then
5:      $q_n = -q_n$ 
6:   end if
7: end for

```

---



Inputs	Symbol	Unit	Contents
Epochs	$t_G$	GPS seconds	Epochs of gradiometer
Epochs	$t_{S_1}, t_{S_2}, t_{S_3}$	GPS seconds	Epochs of attitude of three star trackers
Attitude quaternions	$q_{SRF_1}^{IRF}, q_{SRF_2}^{IRF}, q_{SRF_3}^{IRF}$	unitless	Orientation of star tracker in IRF
Flags	$f_{BBO_1}, f_{BBO_2}, f_{BBO_3}$	unitless	Big-bright-object (BBO) flag of star trackers: 1 = BBO in field of view, 0 = no BBO in field of view
Flags	$f_{VAL_1}, f_{VAL_2}, f_{VAL_3}$	unitless	Valid flag of star trackers: 1 = valid attitude, 0 = invalid attitude
Epochs	$t_{T_1}, t_{T_2}, t_{T_3}$	GPS seconds	Epochs of temperature of three star trackers
Temperatures	$T_{S_1}, T_{S_2}, T_{S_3}$	$^{\circ}C$	Temperature of three star trackers
Half window width	$\Delta t_q$	seconds	Window width for selection of quaternions
Half window width	$\Delta t_T$	seconds	Window width for selection of temperatures
Outputs	Symbol	Unit	Contents
Attitude quaternions	$q_{SRF_1}^{IRF,res}, q_{SRF_2}^{IRF,res}, q_{SRF_3}^{IRF,res}$	unitless	Orientation of $SRF_1, SRF_2, SRF_3$ wrt. IRF resampled to gradiometer epochs
Temperatures	$T_{S_1}^{res}, T_{S_2}^{res}, T_{S_3}^{res}$	$^{\circ}C$	Temperature of three star trackers resampled to gradiometer epochs
Flags	$f_{S_1}^{res}, f_{S_2}^{res}, f_{S_3}^{res}$	unitless	Flags of resampled star tracker data: 1 = usable, 0 = not usable

Table 5: List of inputs and outputs of star tracker preprocessing algorithm



---

**Algorithm 5** Resampling of star tracker quaternions to epochs of gradiometer (part 1)
 

---

```

1: for  $i \leftarrow 1, 3$  do ▷ Loop over three star trackers
2:   Remove all epochs from variables  $\mathbf{t}_{S_i}$ ,  $\mathbf{q}_{SRRF_i}^{IRF}$  and  $\mathbf{f}_{VAL,i}$  for which  $\mathbf{f}_{BBO,i} = 1$ 
3:   Remove all epochs from variables  $\mathbf{t}_{S_i}$  and  $\mathbf{q}_{SRRF_i}^{IRF}$  for which  $\mathbf{f}_{VAL,i} = 0$ 
4:    $\mathbf{q}_{SRRF_1}^{IRF,res} = \text{zeros}(\text{length}(\mathbf{t}_G), 4)$ 
5:   if  $\text{empty}(\mathbf{t}_{S_i})$  then ▷ If no quaternions are available from this star tracker,
6:      $\mathbf{f}_{S_i}^{res} = \text{zeros}(\text{size}(\mathbf{t}_G))$  ▷ set flag of resampled star tracker data to not usable
7:   else
8:      $\mathbf{f}_{S_i}^{res} = \text{ones}(\text{size}(\mathbf{t}_G))$  ▷ Initialise flags of resampled star tracker data
9:      $k = 1$ 
10:     $m = 1$ 
11:    for  $n \leftarrow 1, \text{length}(\mathbf{t}_G)$  do
12:      while  $k < \text{length}(\mathbf{t}_{S_i})$  and  $t_{S_i,k} < t_{G,n} - \Delta t_q$  do
13:         $k = k + 1$ 
14:      end while
15:      while  $m < \text{length}(\mathbf{t}_{S_i})$  and  $t_{S_i,m+1} < t_{G,n} + \Delta t_q$  do
16:         $m = m + 1$ 
17:      end while
18:      if  $m - k \geq 2$  and  $t_{S_i,k} < t_{G,n}$  and  $t_{S_i,m} > t_{G,n}$  then
19:        
$$\boldsymbol{\tau} = \frac{1}{\Delta t_q} \begin{bmatrix} t_{S_i,k} - t_{G,n} \\ \vdots \\ t_{S_i,m} - t_{G,n} \end{bmatrix}$$

20:        
$$\mathbf{A} = \begin{bmatrix} \boldsymbol{\tau}^0 & \boldsymbol{\tau}^1 & \boldsymbol{\tau}^2 \end{bmatrix}$$

21:        
$$\mathbf{y} = \begin{bmatrix} (\mathbf{q}_{SRRF_i,k}^{IRF})^T \\ \vdots \\ (\mathbf{q}_{SRRF_i,m}^{IRF})^T \end{bmatrix}$$

22:        
$$\mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{y}$$

23:        
$$\mathbf{q}_{SRRF_i,n}^{IRF,res} = \begin{bmatrix} \mathbf{x}_{1,1} & \cdots & \mathbf{x}_{1,4} \end{bmatrix}^T$$

24:      else
25:         $\mathbf{f}_{S_i,n}^{res} = 0$ 
26:      end if
27:    end for
28:  end if

```

---

---

**Algorithm 5** Resampling of star tracker quaternions to epochs of gradiometer (part 2)
 

---

```

29:    $\mathbf{T}_{S_i} = \text{zeros}(\text{length}(\mathbf{t}_G), 1)$ 
30:   if  $\text{empty}(\mathbf{t}_{T_i})$  then           ▷ If no temperatures are available from this star tracker,
31:        $\mathbf{f}_{S_i}^{res} = \text{zeros}(\text{size}(\mathbf{t}_G))$        ▷ set flag of resampled star tracker data to not usable
32:   else
33:        $k = 1$ 
34:        $m = 1$ 
35:       for  $n \leftarrow 1, \text{length}(\mathbf{t}_G)$  do
36:           while  $k < \text{length}(\mathbf{t}_{T_i})$  and  $t_{T_i,k} < t_{G,n} - \Delta t_T$  do
37:                $k = k + 1$ 
38:           end while
39:           while  $m < \text{length}(\mathbf{t}_{T_i})$  and  $t_{T_i,m+1} < t_{G,n} + \Delta t_T$  do
40:                $m = m + 1$ 
41:           end while
42:           if  $m - k \geq 2$  and  $t_{T_i,k} < t_{G,n}$  and  $t_{T_i,m} > t_{G,n}$  then
43:                $T_{S_i,n}^{res} = \text{mean}([T_{S_i,k} \ \cdots \ T_{S_i,m}])$ 
44:           else
45:                $f_{S_i,n}^{res} = 0$ 
46:           end if
47:       end for
48:   end if
49: end for

```

---



Inputs	Symbol	Unit	Contents	Source algorithm
Attitude quaternions	$\mathbf{q}_{SRF_1}^{IRF,res}, \mathbf{q}_{SRF_2}^{IRF,res}, \mathbf{q}_{SRF_3}^{IRF,res}$	unitless	Resampled orientation of $SRF_1, SRF_2, SRF_3$ wrt. IRF	Star tracker pre-processing
Flag	$f_{S_1,n}^{res}, f_{S_2,n}^{res}, f_{S_3,n}^{res}$	unitless	Flag of resampled star tracker quaternion: 1 = valid, 0 = invalid	Star tracker pre-processing
Temperature	$T_{S_1,n}^{res}, T_{S_2,n}^{res}, T_{S_3,n}^{res}$	$^{\circ}C$	Resampled temperature of star trackers	Star tracker pre-processing
Constants	Symbol	Unit	Contents	Source
Biases	$\mathbf{b}_{S_1}^{CRF}, \mathbf{b}_{S_2}^{CRF}, \mathbf{b}_{S_3}^{CRF}$	unitless	Star tracker attitude bias in CRF	Star tracker calibration
Rotation	$\mathbf{R}_{SRF_1}^{CRF}, \mathbf{R}_{SRF_2}^{CRF}, \mathbf{R}_{SRF_3}^{CRF}$	unitless	Star tracker orientation in CRF	
Outputs	Symbol	Unit	Contents	
Attitude quaternions	$\mathbf{q}_{CRF}^{IRF}$	unitless	Orientation of CRF wrt. IRF	
Flags	$\mathbf{f}$	unitless	1 = valid, 0 = invalid	
Square-sum of residuals	$\Omega$	$rad^2$	Square-sum of residuals	
Cofactor matrices	$\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3, \mathbf{Q}_{12}, \mathbf{Q}_{13}, \mathbf{Q}_{23}, \mathbf{Q}_{123}$	unitless	Cofactor matrices	

Table 6: List of inputs and outputs of star tracker combination algorithm

---

**Algorithm 6** Star tracker combination (part 1)
 

---

- 1:  $\mathbf{P}^{SRF} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{1}{10^2} \end{bmatrix}$  ▷ attitude around boresight axis 10 times worse than others
  - 2:  $\mathbf{P}_1^{CRF} = \mathbf{R}_{SRF_1}^{CRF} * \mathbf{P}^{SRF} * (\mathbf{R}_{SRF_1}^{CRF})^T$
  - 3:  $\mathbf{P}_2^{CRF} = \mathbf{R}_{SRF_2}^{CRF} * \mathbf{P}^{SRF} * (\mathbf{R}_{SRF_2}^{CRF})^T$
  - 4:  $\mathbf{P}_3^{CRF} = \mathbf{R}_{SRF_3}^{CRF} * \mathbf{P}^{SRF} * (\mathbf{R}_{SRF_3}^{CRF})^T$
  - 5: Convert  $\mathbf{R}_{SRF_1}^{CRF}$  to  $q_{SRF_1}^{CRF}$  using Algorithm 3
  - 6: Convert  $\mathbf{R}_{SRF_2}^{CRF}$  to  $q_{SRF_2}^{CRF}$  using Algorithm 3
  - 7: Convert  $\mathbf{R}_{SRF_3}^{CRF}$  to  $q_{SRF_3}^{CRF}$  using Algorithm 3
  - 8:  $\mathbf{e}_1^{CRF} = \text{zeros}(N, 3)$
  - 9:  $\mathbf{e}_2^{CRF} = \text{zeros}(N, 3)$
  - 10:  $\mathbf{e}_3^{CRF} = \text{zeros}(N, 3)$
  - 11:  $\mathbf{q}_{CRF}^{IRF} = \text{zeros}(N, 4)$
  - 12: **for**  $n \leftarrow 1, N$  **do**
  - 13:  $\mathbf{b}_{S_{1,n}}^{CRF} = \mathbf{b}_{C_1}^{CRF} + T_{S_{1,n}}^{res} \mathbf{b}_{T_1}^{CRF}$
  - 14:  $\mathbf{b}_{S_{2,n}}^{CRF} = \mathbf{b}_{C_2}^{CRF} + T_{S_{2,n}}^{res} \mathbf{b}_{T_2}^{CRF}$
  - 15:  $\mathbf{b}_{S_{3,n}}^{CRF} = \mathbf{b}_{C_3}^{CRF} + T_{S_{3,n}}^{res} \mathbf{b}_{T_3}^{CRF}$
  - 16:  $\mathbf{b}_{S_{1,n}}^{SRF_1} = \mathbf{R}_{CRF}^{SRF_1} \mathbf{b}_{S_{1,n}}^{CRF}$
  - 17:  $\mathbf{b}_{S_{2,n}}^{SRF_2} = \mathbf{R}_{CRF}^{SRF_2} \mathbf{b}_{S_{2,n}}^{CRF}$
  - 18:  $\mathbf{b}_{S_{3,n}}^{SRF_3} = \mathbf{R}_{CRF}^{SRF_3} \mathbf{b}_{S_{3,n}}^{CRF}$
  - 19:  $q_{12,n} = (q_{SRF_1,n}^{IRF,res} q_{SRF_1}^{SRF_1}) * (q_{SRF_2,n}^{IRF,res} q_{SRF_2}^{SRF_2})$  ▷ quaternion multiplication/conjugation
  - 20:  $q_{13,n} = (q_{SRF_1,n}^{IRF,res} q_{SRF_1}^{SRF_1}) * (q_{SRF_3,n}^{IRF,res} q_{SRF_3}^{SRF_3})$  ▷ quaternion multiplication/conjugation
  - 21:  $q_{23,n} = (q_{SRF_2,n}^{IRF,res} q_{SRF_2}^{SRF_2}) * (q_{SRF_3,n}^{IRF,res} q_{SRF_3}^{SRF_3})$  ▷ quaternion multiplication/conjugation
  - 22:  $\mathbf{d}_{12,n} = 2 * \text{sign } q_{12,n,0} \begin{bmatrix} q_{12,n,1} \\ q_{12,n,2} \\ q_{12,n,3} \end{bmatrix} + \mathbf{b}_{S_{1,n}}^{CRF} - \mathbf{b}_{S_{2,n}}^{CRF}$
  - 23:  $\mathbf{d}_{13,n} = 2 * \text{sign } q_{13,n,0} \begin{bmatrix} q_{13,n,1} \\ q_{13,n,2} \\ q_{13,n,3} \end{bmatrix} + \mathbf{b}_{S_{1,n}}^{CRF} - \mathbf{b}_{S_{3,n}}^{CRF}$
  - 24:  $\mathbf{d}_{23,n} = 2 * \text{sign } q_{23,n,0} \begin{bmatrix} q_{23,n,1} \\ q_{23,n,2} \\ q_{23,n,3} \end{bmatrix} + \mathbf{b}_{S_{2,n}}^{CRF} - \mathbf{b}_{S_{3,n}}^{CRF}$
  - 25: **end for**
-

---

**Algorithm 6** Star tracker combination (part 2)
 

---

```

26: for  $n \leftarrow 1, N$  do
27:   if  $f_{S_{1,n}}^{res} = 1$  and  $f_{S_{2,n}}^{res} = 1$  and  $f_{S_{3,n}}^{res} = 0$  then
28:      $\mathbf{e}_{1,n}^{CRF} = -(\mathbf{P}_1^{CRF} + \mathbf{P}_2^{CRF})^{-1} \mathbf{P}_2^{CRF} \mathbf{d}_{12,n}$ 
29:      $\mathbf{e}_{2,n}^{CRF} = \mathbf{e}_{1,n}^{CRF} + \mathbf{d}_{12,n}$ 
30:   end if
31:   if  $f_{S_{1,n}}^{res} = 1$  and  $f_{S_{2,n}}^{res} = 0$  and  $f_{S_{3,n}}^{res} = 1$  then
32:      $\mathbf{e}_{1,n}^{CRF} = -(\mathbf{P}_1^{CRF} + \mathbf{P}_3^{CRF})^{-1} \mathbf{P}_3^{CRF} \mathbf{d}_{13,n}$ 
33:      $\mathbf{e}_{3,n}^{CRF} = \mathbf{e}_{1,n}^{CRF} + \mathbf{d}_{13,n}$ 
34:   end if
35:   if  $f_{S_{1,n}}^{res} = 0$  and  $f_{S_{2,n}}^{res} = 1$  and  $f_{S_{3,n}}^{res} = 1$  then
36:      $\mathbf{e}_{2,n}^{CRF} = -(\mathbf{P}_2^{CRF} + \mathbf{P}_3^{CRF})^{-1} \mathbf{P}_3^{CRF} \mathbf{d}_{23,n}$ 
37:      $\mathbf{e}_{3,n}^{CRF} = \mathbf{e}_{2,n}^{CRF} + \mathbf{d}_{23,n}$ 
38:   end if
39:   if  $f_{S_{1,n}}^{res} = 1$  and  $f_{S_{2,n}}^{res} = 1$  and  $f_{S_{3,n}}^{res} = 1$  then
40:      $\mathbf{e}_{1,n}^{CRF} = -(\mathbf{P}_1^{CRF} + \mathbf{P}_2^{CRF} + \mathbf{P}_3^{CRF})^{-1} (\mathbf{P}_2^{CRF} \mathbf{d}_{12,n} + \mathbf{P}_3^{CRF} \mathbf{d}_{13,n})$ 
41:      $\mathbf{e}_{2,n}^{CRF} = \mathbf{e}_{1,n}^{CRF} + \mathbf{d}_{12,n}$ 
42:      $\mathbf{e}_{3,n}^{CRF} = \mathbf{e}_{1,n}^{CRF} + \mathbf{d}_{13,n}$ 
43:   end if
44:   if  $f_{S_{1,n}}^{res} = 1$  or  $f_{S_{2,n}}^{res} = 1$  or  $f_{S_{3,n}}^{res} = 1$  then           ▷ set flag of combined quaternion
45:      $f_n = 1$ 
46:   else
47:      $f_n = 0$ 
48:   end if
49:   if  $f_{S_{1,n}}^{res} = 1$  then
50:      $q_{CRF,n}^{IRF} = (q_{SRF_{1,n}}^{IRF} \begin{bmatrix} 1 \\ -(\mathbf{e}_{1,n}^{SRF_1} + \mathbf{b}_{1,n}^{SRF_1})/2 \end{bmatrix}) q_{CRF}^{SRF_1}$            ▷ quat. multiplications
51:   end if
52:   if  $f_{S_{2,n}}^{res} = 1$  then
53:      $q_{CRF,n}^{IRF} = (q_{SRF_{2,n}}^{IRF} \begin{bmatrix} 1 \\ -(\mathbf{e}_{2,n}^{SRF_2} + \mathbf{b}_{2,n}^{SRF_2})/2 \end{bmatrix}) q_{CRF}^{SRF_2}$            ▷ quat. multiplications
54:   end if
55:   if  $f_{S_{3,n}}^{res} = 1$  then
56:      $q_{CRF,n}^{IRF} = (q_{SRF_{3,n}}^{IRF} \begin{bmatrix} 1 \\ -(\mathbf{e}_{3,n}^{SRF_3} + \mathbf{b}_{3,n}^{SRF_3})/2 \end{bmatrix}) q_{CRF}^{SRF_3}$            ▷ quat. multiplications
57:   end if
58:    $q_{CRF,n}^{IRF} = \frac{q_{CRF,n}^{IRF}}{|q_{CRF,n}^{IRF}|}$            ▷ normalize quaternion
59: end for

```

---

---

**Algorithm 6** Star tracker combination (part 3, optional)
 

---

```

60:  $\mathbf{R}^{SRF} = \text{chol}(\mathbf{P}^{SRF})$                                 ▷ Cholesky factorization
61:  $\Omega = 0$                                                     ▷ square-sum of residuals
62: for  $n \leftarrow 1, N$  do
63:    $\mathbf{e}_{1,n}^{SRF_1} = \mathbf{R}_{CRF}^{SRF_1} \mathbf{e}_{1,n}^{CRF}$ 
64:    $\mathbf{e}_{2,n}^{SRF_2} = \mathbf{R}_{CRF}^{SRF_2} \mathbf{e}_{2,n}^{CRF}$ 
65:    $\mathbf{e}_{3,n}^{SRF_3} = \mathbf{R}_{CRF}^{SRF_3} \mathbf{e}_{3,n}^{CRF}$ 
66:    $\Omega = \Omega + (\mathbf{R}^{SRF} \mathbf{e}_{1,n}^{SRF_1})^T (\mathbf{R}^{SRF} \mathbf{e}_{1,n}^{SRF_1})$ 
67:    $\Omega = \Omega + (\mathbf{R}^{SRF} \mathbf{e}_{2,n}^{SRF_2})^T (\mathbf{R}^{SRF} \mathbf{e}_{2,n}^{SRF_2})$ 
68:    $\Omega = \Omega + (\mathbf{R}^{SRF} \mathbf{e}_{3,n}^{SRF_3})^T (\mathbf{R}^{SRF} \mathbf{e}_{3,n}^{SRF_3})$ 
69: end for
70:  $\mathbf{Q}_1 = (\mathbf{P}_1^{CRF})^{-1}$                                         ▷ cofactor matrix
71:  $\mathbf{Q}_2 = (\mathbf{P}_2^{CRF})^{-1}$                                         ▷ cofactor matrix
72:  $\mathbf{Q}_3 = (\mathbf{P}_3^{CRF})^{-1}$                                         ▷ cofactor matrix
73:  $\mathbf{Q}_{12} = (\mathbf{P}_1^{CRF} + \mathbf{P}_2^{CRF})^{-1}$                         ▷ cofactor matrix
74:  $\mathbf{Q}_{13} = (\mathbf{P}_1^{CRF} + \mathbf{P}_3^{CRF})^{-1}$                         ▷ cofactor matrix
75:  $\mathbf{Q}_{23} = (\mathbf{P}_2^{CRF} + \mathbf{P}_3^{CRF})^{-1}$                         ▷ cofactor matrix
76:  $\mathbf{Q}_{123} = (\mathbf{P}_1^{CRF} + \mathbf{P}_2^{CRF} + \mathbf{P}_3^{CRF})^{-1}$         ▷ cofactor matrix

```

---



## 5 Adjusting the misalignment between star trackers and gradiometer

The calibration against the angular rates from the combined star trackers and the gravity field model yields not only calibration parameters for the gradiometer (inverse calibration matrices and quadratic factor matrices), but also the misalignment between star trackers and gradiometer. This misalignment needs to be corrected prior to the angular rate reconstruction. The misalignment parameters are small angles denoted by  $\alpha$ ,  $\beta$  and  $\gamma$ . They are specified at two reference epochs  $t_a$  and  $t_b$ , between which they have to be linearly interpolated. The misalignment between the star trackers and the gradiometer is taken into account by applying a small rotation to the combined star sensor quaternion  $\mathbf{q}_{IRF}^{CRF}$  that connects the inertial reference frame with the common reference frame of the star trackers.

Inputs	Symbol	Unit	Contents	Source algorithm
Epochs	$t_a, t_b$	GPS second	Reference epochs for interpolation	Gradiometer calibration
Attitude quaternions	$\mathbf{q}_{IRF}^{CRF}$	unitless	Orientation of $CRF$ wrt. IRF	Star tracker combination
Epochs	$t$	GPS second	Epoch of $\mathbf{q}_{IRF}^{CRF}$	Star tracker combination
Misalignments	$\alpha_a, \beta_a, \gamma_a, \alpha_b, \beta_b, \gamma_b$	radians	Misalignment between CRF and GRF at epochs $t_a$ and $t_b$	Gradiometer calibration
Outputs	Symbol	Unit	Contents	
Attitude quaternions	$\mathbf{q}_{IRF}^{GRF}$	unitless	Orientation of GRF wrt. IRF	

Table 7: List of inputs and outputs of algorithm for adjusting the misalignment between star trackers and gradiometer

---

### Algorithm 7 Adjustment of the misalignment between star trackers and gradiometer

---

- 1: **for**  $n \leftarrow 1, N$  **do**
  - 2:      $\alpha_n = \frac{t_b - t_n}{t_b - t_a} \alpha_a + \frac{t_n - t_a}{t_b - t_a} \alpha_b$       $\triangleright$  Linear interpolation of misalignments
  - 3:      $\beta_n = \frac{t_b - t_n}{t_b - t_a} \beta_a + \frac{t_n - t_a}{t_b - t_a} \beta_b$
  - 4:      $\gamma_n = \frac{t_b - t_n}{t_b - t_a} \gamma_a + \frac{t_n - t_a}{t_b - t_a} \gamma_b$
  - 5:      $q_{CRF,n}^{GRF} = \left[ 1 \quad -\alpha_n/2 \quad -\beta_n/2 \quad -\gamma_n/2 \right]^T / \sqrt{1 + \frac{1}{4}(\alpha_n^2 + \beta_n^2 + \gamma_n^2)}$
  - 6:      $q_{IRF,n}^{GRF} = q_{IRF,n}^{CRF} q_{CRF,n}^{GRF}$       $\triangleright$  Quaternion multiplication
  - 7: **end for**
-





## 6 Accelerometer data processing

### 6.1 Accelerometer data calibration

The relationship of the measured and true acceleration is defined by the quadratic function

$$\hat{\mathbf{a}}_i = \hat{\mathbf{b}}_i + \mathbf{M}_i \mathbf{a}_i + \mathbf{K}_i \mathbf{a}_i^2 + \mathbf{W}_i \dot{\boldsymbol{\omega}} + \hat{\mathbf{n}}_i \quad (45)$$

where  $\hat{\mathbf{a}}_i$  is the measured acceleration,  $\hat{\mathbf{b}}_i$  is the bias of the measured acceleration,  $\mathbf{M}_i$  is a calibration matrix for the  $i$ -th accelerometer,  $\mathbf{a}_i$  is the true acceleration,  $\mathbf{K}_i$  is the quadratic factor matrix,  $\mathbf{W}_i$  is the angular acceleration coupling matrix,  $\dot{\boldsymbol{\omega}}$  is the true angular acceleration, and  $\hat{\mathbf{n}}_i$  is noise in the measured acceleration. It should be noted that  $\mathbf{M}_i$  is a general  $3 \times 3$  matrix and  $\mathbf{K}_i$  is a  $3 \times 3$  diagonal matrix. The elements of  $\mathbf{W}_i$  depend on the onboard proof mass control and are defined as

$$\mathbf{W}_i = \begin{bmatrix} 0 & 0 & 0 \\ e_i & 0 & g_i \\ 0 & f_i & 0 \end{bmatrix} \quad \text{for } i \in \{1, 4\}, \quad (46)$$

$$\mathbf{W}_i = \begin{bmatrix} 0 & 0 & g_i \\ 0 & 0 & 0 \\ e_i & f_i & 0 \end{bmatrix} \quad \text{for } i \in \{2, 5\} \quad (47)$$

and

$$\mathbf{W}_i = \begin{bmatrix} 0 & f_i & 0 \\ e_i & 0 & g_i \\ 0 & 0 & 0 \end{bmatrix} \quad \text{for } i \in \{3, 6\}. \quad (48)$$

For the square of a vector as in  $\mathbf{a}_i^2$ , we use the convention that the elements of the vector are squared, i.e.

$$\mathbf{a}_i^2 = \begin{bmatrix} a_{ix}^2 \\ a_{iy}^2 \\ a_{iz}^2 \end{bmatrix}. \quad (49)$$

Differential and common mode acceleration are defined by

$$\begin{bmatrix} \mathbf{a}_{dij} \\ \mathbf{a}_{cij} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \mathbf{a}_i - \mathbf{a}_j \\ \mathbf{a}_i + \mathbf{a}_j \end{bmatrix}. \quad (50)$$

When defining the inverse calibration matrix as

$$\mathbf{M}_{ij} = 2 \begin{bmatrix} \mathbf{M}_i + \mathbf{M}_j & \mathbf{M}_i - \mathbf{M}_j \\ \mathbf{M}_i - \mathbf{M}_j & \mathbf{M}_i + \mathbf{M}_j \end{bmatrix}^{-1}, \quad (51)$$

we can reformulate Eq. (45) to

$$\begin{bmatrix} \hat{\mathbf{a}}_{dij} \\ \hat{\mathbf{a}}_{cij} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{b}}_{dij} \\ \hat{\mathbf{b}}_{cij} \end{bmatrix} + \mathbf{M}_{ij}^{-1} \begin{bmatrix} \mathbf{a}_{dij} \\ \mathbf{a}_{cij} \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \begin{bmatrix} \mathbf{a}_i^2 \\ \mathbf{a}_j^2 \end{bmatrix} + \begin{bmatrix} \mathbf{W}_{dij} \\ \mathbf{W}_{cij} \end{bmatrix} \dot{\boldsymbol{\omega}} + \begin{bmatrix} \hat{\mathbf{n}}_{dij} \\ \hat{\mathbf{n}}_{cij} \end{bmatrix}, \quad (52)$$

where all differential and common terms are signified by subscripts  $d$  and  $c$ , respectively, and defined analogously to Eq. (50).

A so-called satellite shaking procedure yields a first estimate of the inverse calibration matrix  $\mathbf{M}_{ij}$ , which we denote by  $\hat{\mathbf{M}}_{ij}$ . Applying the first estimate of the inverse calibration matrix to the measured acceleration yields

$$\begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} = \hat{\mathbf{M}}_{ij} \begin{bmatrix} \hat{\mathbf{a}}_{dij} \\ \hat{\mathbf{a}}_{cij} \end{bmatrix}, \quad (53)$$

where  $\bar{\mathbf{a}}_{dij}$  and  $\bar{\mathbf{a}}_{cij}$  are calibrated differential and common mode acceleration, respectively, of the first stage of the calibration. We regard  $\bar{\mathbf{a}}_{dij}$  and  $\bar{\mathbf{a}}_{cij}$  as good approximations of  $\mathbf{a}_{dij}$  and  $\mathbf{a}_{cij}$ , respectively, which will be refined in the second stage of the calibration.

Inserting Eq. (53) into Eq. (52) gives

$$\begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \hat{\mathbf{M}}_{ij} \mathbf{M}_{ij}^{-1} \begin{bmatrix} \mathbf{a}_{dij} \\ \mathbf{a}_{cij} \end{bmatrix} + \frac{1}{2} \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \begin{bmatrix} \mathbf{a}_i^2 \\ \mathbf{a}_j^2 \end{bmatrix} + \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{W}_{dij} \\ \mathbf{W}_{cij} \end{bmatrix} \dot{\boldsymbol{\omega}} + \begin{bmatrix} \bar{\mathbf{n}}_{dij} \\ \bar{\mathbf{n}}_{cij} \end{bmatrix}, \quad (54)$$

where  $\bar{\mathbf{b}}_{dij}$ ,  $\bar{\mathbf{b}}_{cij}$ ,  $\bar{\mathbf{n}}_{dij}$ , and  $\bar{\mathbf{n}}_{cij}$  are defined analogously to Eq. (53). In order to proceed, we need to find approximations for  $\mathbf{a}_i^2$  and  $\mathbf{a}_j^2$ . For this purpose, we assume that quadratic factors and angular acceleration couplings are small, i.e.  $\mathbf{K}_i \approx \mathbf{0}$ ,  $\mathbf{K}_j \approx \mathbf{0}$ ,  $\mathbf{W}_{dij} \approx \mathbf{0}$  and  $\mathbf{W}_{cij} \approx \mathbf{0}$ , the inverse calibration matrix estimated in the satellite shaking procedure approximates the true inverse calibration matrix well, i.e.  $\hat{\mathbf{M}}_{ij} \approx \mathbf{M}_{ij}$ , and that we can neglect the noise terms, i.e.  $\bar{\mathbf{n}}_{dij} \approx \mathbf{0}$  and  $\bar{\mathbf{n}}_{cij} \approx \mathbf{0}$ . With these assumptions, Eq. (54) reduces to

$$\begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \begin{bmatrix} \mathbf{a}_{dij} \\ \mathbf{a}_{cij} \end{bmatrix}, \quad (55)$$

which leads to

$$\begin{bmatrix} \mathbf{a}_i^2 \\ \mathbf{a}_j^2 \end{bmatrix} = \begin{bmatrix} (\bar{\mathbf{a}}_i - \bar{\mathbf{b}}_i)^2 \\ (\bar{\mathbf{a}}_j - \bar{\mathbf{b}}_j)^2 \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{a}}_i^2 \\ \bar{\mathbf{a}}_j^2 \end{bmatrix} - 2 \begin{bmatrix} \bar{\mathbf{a}}_i \bar{\mathbf{b}}_i \\ \bar{\mathbf{a}}_j \bar{\mathbf{b}}_j \end{bmatrix} + \begin{bmatrix} \bar{\mathbf{b}}_i^2 \\ \bar{\mathbf{b}}_j^2 \end{bmatrix}. \quad (56)$$

Inserting this result for  $\mathbf{a}_i^2$  and  $\mathbf{a}_j^2$  into Eq. (54) gives

$$\begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \hat{\mathbf{M}}_{ij} \mathbf{M}_{ij}^{-1} \begin{bmatrix} \mathbf{a}_{dij} \\ \mathbf{a}_{cij} \end{bmatrix} + \frac{1}{2} \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \left( \begin{bmatrix} \bar{\mathbf{a}}_i^2 \\ \bar{\mathbf{a}}_j^2 \end{bmatrix} - 2 \begin{bmatrix} \bar{\mathbf{a}}_i \bar{\mathbf{b}}_i \\ \bar{\mathbf{a}}_j \bar{\mathbf{b}}_j \end{bmatrix} + \begin{bmatrix} \bar{\mathbf{b}}_i^2 \\ \bar{\mathbf{b}}_j^2 \end{bmatrix} \right) + \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{W}_{dij} \\ \mathbf{W}_{cij} \end{bmatrix} \dot{\boldsymbol{\omega}} + \begin{bmatrix} \bar{\mathbf{n}}_{dij} \\ \bar{\mathbf{n}}_{cij} \end{bmatrix}. \quad (57)$$

In order to simplify the equation, we note that

$$\begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{a}}_i \bar{\mathbf{b}}_i \\ \bar{\mathbf{a}}_j \bar{\mathbf{b}}_j \end{bmatrix} = \begin{bmatrix} \mathbf{K}_i \bar{\mathbf{b}}_i & -\mathbf{K}_j \bar{\mathbf{b}}_j \\ \mathbf{K}_i \bar{\mathbf{b}}_i & \mathbf{K}_j \bar{\mathbf{b}}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{a}}_i \\ \bar{\mathbf{a}}_j \end{bmatrix} \quad (58)$$

$$= \begin{bmatrix} \mathbf{K}_i \bar{\mathbf{b}}_i & -\mathbf{K}_j \bar{\mathbf{b}}_j \\ \mathbf{K}_i \bar{\mathbf{b}}_i & \mathbf{K}_j \bar{\mathbf{b}}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{a}}_{cij} + \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} - \bar{\mathbf{a}}_{dij} \end{bmatrix} \quad (59)$$

$$= \begin{bmatrix} \mathbf{K}_i \bar{\mathbf{b}}_i + \mathbf{K}_j \bar{\mathbf{b}}_j & \mathbf{K}_i \bar{\mathbf{b}}_i - \mathbf{K}_j \bar{\mathbf{b}}_j \\ \mathbf{K}_i \bar{\mathbf{b}}_i - \mathbf{K}_j \bar{\mathbf{b}}_j & \mathbf{K}_i \bar{\mathbf{b}}_i + \mathbf{K}_j \bar{\mathbf{b}}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix}, \quad (60)$$

which leads to

$$\begin{aligned} \begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} &= \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \frac{1}{2} \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{b}}_i^2 \\ \bar{\mathbf{b}}_j^2 \end{bmatrix} + \hat{\mathbf{M}}_{ij} \mathbf{M}_{ij}^{-1} \begin{bmatrix} \mathbf{a}_{dij} \\ \mathbf{a}_{cij} \end{bmatrix} \\ &\quad - \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{K}_i \bar{\mathbf{b}}_i + \mathbf{K}_j \bar{\mathbf{b}}_j & \mathbf{K}_i \bar{\mathbf{b}}_i - \mathbf{K}_j \bar{\mathbf{b}}_j \\ \mathbf{K}_i \bar{\mathbf{b}}_i - \mathbf{K}_j \bar{\mathbf{b}}_j & \mathbf{K}_i \bar{\mathbf{b}}_i + \mathbf{K}_j \bar{\mathbf{b}}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} \\ &\quad + \frac{1}{2} \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{a}}_i^2 \\ \bar{\mathbf{a}}_j^2 \end{bmatrix} + \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{W}_{dij} \\ \mathbf{W}_{cij} \end{bmatrix} \dot{\boldsymbol{\omega}} + \begin{bmatrix} \bar{\mathbf{n}}_{dij} \\ \bar{\mathbf{n}}_{cij} \end{bmatrix}. \end{aligned} \quad (61)$$

We solve this equation for  $\mathbf{a}_{dij}$  and  $\mathbf{a}_{cij}$ , which gives

$$\begin{aligned} \begin{bmatrix} \mathbf{a}_{dij} \\ \mathbf{a}_{cij} \end{bmatrix} &= -\mathbf{M}_{ij} \hat{\mathbf{M}}_{ij}^{-1} \left( \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \frac{1}{2} \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{b}}_i^2 \\ \bar{\mathbf{b}}_j^2 \end{bmatrix} \right) \\ &\quad + \mathbf{M}_{ij} \left( \hat{\mathbf{M}}_{ij}^{-1} + \begin{bmatrix} \mathbf{K}_i \bar{\mathbf{b}}_i + \mathbf{K}_j \bar{\mathbf{b}}_j & \mathbf{K}_i \bar{\mathbf{b}}_i - \mathbf{K}_j \bar{\mathbf{b}}_j \\ \mathbf{K}_i \bar{\mathbf{b}}_i - \mathbf{K}_j \bar{\mathbf{b}}_j & \mathbf{K}_i \bar{\mathbf{b}}_i + \mathbf{K}_j \bar{\mathbf{b}}_j \end{bmatrix} \right) \begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} \\ &\quad - \frac{1}{2} \mathbf{M}_{ij} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{a}}_i^2 \\ \bar{\mathbf{a}}_j^2 \end{bmatrix} - \mathbf{M}_{ij} \begin{bmatrix} \mathbf{W}_{dij} \\ \mathbf{W}_{cij} \end{bmatrix} \dot{\boldsymbol{\omega}} - \mathbf{M}_{ij} \hat{\mathbf{M}}_{ij}^{-1} \begin{bmatrix} \bar{\mathbf{n}}_{dij} \\ \bar{\mathbf{n}}_{cij} \end{bmatrix}. \end{aligned} \quad (62)$$

This equation shows that  $\mathbf{a}_{dij}$  and  $\mathbf{a}_{cij}$  are a quadratic function of  $\bar{\mathbf{a}}_{dij}$  and  $\bar{\mathbf{a}}_{cij}$  plus a linear function of  $\dot{\boldsymbol{\omega}}$ . We rewrite the equation as

$$\begin{bmatrix} \mathbf{a}_{dij} \\ \mathbf{a}_{cij} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \bar{\mathbf{M}}_{ij} \begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} + \bar{\mathbf{K}}_{ij} \begin{bmatrix} \bar{\mathbf{a}}_i^2 \\ \bar{\mathbf{a}}_j^2 \end{bmatrix} + \bar{\mathbf{W}}_{ij} \dot{\boldsymbol{\omega}} + \begin{bmatrix} \bar{\mathbf{n}}_{dij} \\ \bar{\mathbf{n}}_{cij} \end{bmatrix} \quad (63)$$

where

$$\begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} = -\mathbf{M}_{ij} \hat{\mathbf{M}}_{ij}^{-1} \left( \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \frac{1}{2} \hat{\mathbf{M}}_{ij} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{b}}_i^2 \\ \bar{\mathbf{b}}_j^2 \end{bmatrix} \right), \quad (64)$$

$$\bar{\mathbf{M}}_{ij} = \mathbf{M}_{ij} \left( \hat{\mathbf{M}}_{ij}^{-1} + \begin{bmatrix} \mathbf{K}_i \bar{\mathbf{b}}_i + \mathbf{K}_j \bar{\mathbf{b}}_j & \mathbf{K}_i \bar{\mathbf{b}}_i - \mathbf{K}_j \bar{\mathbf{b}}_j \\ \mathbf{K}_i \bar{\mathbf{b}}_i - \mathbf{K}_j \bar{\mathbf{b}}_j & \mathbf{K}_i \bar{\mathbf{b}}_i + \mathbf{K}_j \bar{\mathbf{b}}_j \end{bmatrix} \right), \quad (65)$$

$$\bar{\mathbf{K}}_{ij} = -\frac{1}{2}\mathbf{M}_{ij} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix} \approx -\frac{1}{2} \begin{bmatrix} \mathbf{K}_i & -\mathbf{K}_j \\ \mathbf{K}_i & \mathbf{K}_j \end{bmatrix}. \quad (66)$$

and

$$\bar{\mathbf{W}}_{ij} = -\mathbf{M}_{ij} \begin{bmatrix} \mathbf{W}_{dij} \\ \mathbf{W}_{cij} \end{bmatrix} \approx - \begin{bmatrix} \mathbf{W}_{dij} \\ \mathbf{W}_{cij} \end{bmatrix}. \quad (67)$$

The matrices  $\bar{\mathbf{M}}_{ij}$ ,  $\bar{\mathbf{K}}_{ij}$  and  $\bar{\mathbf{W}}_{ij}$  are determined in the calibration against the combined star tracker angular rates and a gravity field model. It should be noted that the diagonal elements of  $\mathbf{M}_{ij}$  are coupled to the accelerometer biases through the quadratic factors, which is a consequence of using the biased acceleration measurements as proxy for the true acceleration. Since the biases are drifting over time, we should expect that the diagonal elements of each  $3 \times 3$  submatrix of  $\bar{\mathbf{M}}_{ij}$  drift in the same way.

Noting that we have to estimate  $\bar{\mathbf{M}}_{ij}$ ,  $\bar{\mathbf{K}}_{ij}$  and  $\bar{\mathbf{W}}_{ij}$  because the true inverse calibration matrix  $\mathbf{M}_{ij}$  is unknown, we replace  $\mathbf{a}_{dij}$  and  $\mathbf{a}_{cij}$  by  $\bar{\mathbf{a}}_{dij}$  and  $\bar{\mathbf{a}}_{cij}$ , respectively, in order to signify that we obtain not the true acceleration, and find the calibrated acceleration

$$\begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \bar{\mathbf{M}}_{ij} \begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} + \bar{\mathbf{K}}_{ij} \begin{bmatrix} (\bar{\mathbf{a}}_{cij} + \bar{\mathbf{a}}_{dij})^2 \\ (\bar{\mathbf{a}}_{cij} - \bar{\mathbf{a}}_{dij})^2 \end{bmatrix} + \bar{\mathbf{W}}_{ij}\dot{\boldsymbol{\omega}} + \begin{bmatrix} \bar{\mathbf{n}}_{dij} \\ \bar{\mathbf{n}}_{cij} \end{bmatrix}. \quad (68)$$

The equation contains the true angular acceleration  $\dot{\boldsymbol{\omega}}$ , which is also unknown. We use the angular acceleration  $\dot{\bar{\boldsymbol{\omega}}}$  as a proxy, which is calculated using Algorithm 14 using as input the combined star tracker quaternions, which are an output of Algorithm 6, and the gradiometer angular accelerations that are calculated from  $\bar{\mathbf{a}}_{dij}$  and  $\bar{\mathbf{a}}_{cij}$  according to Algorithm 12. We thus exchange  $\dot{\boldsymbol{\omega}}$  by  $\dot{\bar{\boldsymbol{\omega}}}$  in Eq.(69) and obtain

$$\begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{b}}_{dij} \\ \bar{\mathbf{b}}_{cij} \end{bmatrix} + \bar{\mathbf{M}}_{ij} \begin{bmatrix} \bar{\mathbf{a}}_{dij} \\ \bar{\mathbf{a}}_{cij} \end{bmatrix} + \bar{\mathbf{K}}_{ij} \begin{bmatrix} (\bar{\mathbf{a}}_{cij} + \bar{\mathbf{a}}_{dij})^2 \\ (\bar{\mathbf{a}}_{cij} - \bar{\mathbf{a}}_{dij})^2 \end{bmatrix} + \bar{\mathbf{W}}_{ij}\dot{\bar{\boldsymbol{\omega}}} + \begin{bmatrix} \bar{\mathbf{n}}_{dij} \\ \bar{\mathbf{n}}_{cij} \end{bmatrix}. \quad (69)$$

Now that we derived the equations for the calibration in detail, we can summarise the algorithm for the gradiometer calibration as follows. In the first stage, we use Eq. (53) to apply the inverse calibration matrices determined in the satellite shaking procedure. In the second stage, we use Eq. (69) to apply the calibration matrices  $\bar{\mathbf{M}}_{ij}$ ,  $\bar{\mathbf{K}}_{ij}$  and  $\bar{\mathbf{W}}_{ij}$ , which are determined in an advanced calibration procedure from science mode and shaking mode data. In both stages, we linearly interpolate the calibration matrices in order to account for small drifts in the calibration parameters. The matrices  $\hat{\mathbf{M}}_{ij}$  are interpolated between two reference epochs  $\hat{t}_a$  and  $\hat{t}_b$  and the matrices  $\bar{\mathbf{M}}_{ij}$ ,  $\bar{\mathbf{K}}_{ij}$  and  $\bar{\mathbf{W}}_{ij}$  are interpolated between two reference epochs  $\bar{t}_a$  and  $\bar{t}_b$ . The reference epochs  $\hat{t}_a$  and  $\hat{t}_b$  refer to the dates of satellite shaking procedures, whereas  $\bar{t}_a$  and  $\bar{t}_b$  are manually selected based on reported onboard events and observed data quality, i.e. the time intervals  $[\hat{t}_a, \hat{t}_b]$  and  $[\bar{t}_a, \bar{t}_b]$  are not necessarily the same.



Inputs	Symbol	Unit	Contents	Source algorithm
Epochs	$\hat{t}_a, \hat{t}_b$	$s$	References epochs in GPS seconds	Satellite shaking procedure
Acceleration	$\mathbf{a}_{d14}, \mathbf{a}_{c14}, \mathbf{a}_{d25}, \mathbf{a}_{c25}, \mathbf{a}_{d36}, \mathbf{a}_{c36}$	$m/s^2$	Measured common and differential mode acceleration	EKG_NOM_1b
Inverse calibration matrices	$\hat{\mathbf{M}}_{14a}, \hat{\mathbf{M}}_{36a}, \hat{\mathbf{M}}_{25a}, \hat{\mathbf{M}}_{14b}, \hat{\mathbf{M}}_{25b}, \hat{\mathbf{M}}_{36b}$	unitless	Inverse calibration matrices at epochs $t_a$ and $t_b$	Satellite shaking procedure
Outputs	Symbol	Unit	Contents	
Acceleration	$\bar{\mathbf{a}}_{d14}, \bar{\mathbf{a}}_{c14}, \bar{\mathbf{a}}_{d25}, \bar{\mathbf{a}}_{c25}, \bar{\mathbf{a}}_{d36}, \bar{\mathbf{a}}_{c36}$	$m/s^2$	Shaking mode calibrated common and differential mode acceleration	

Table 8: List of inputs and outputs of the gradiometer calibration algorithm (satellite shaking)

---

**Algorithm 8** Gradiometer calibration (satellite shaking)

---

```

1: for  $n \leftarrow n_a, n_b$  do
2:   
$$\begin{bmatrix} \bar{\mathbf{a}}_{d14n} \\ \bar{\mathbf{a}}_{c14n} \end{bmatrix} = \left( \frac{\hat{t}_b - t_n}{\hat{t}_b - \hat{t}_a} \hat{\mathbf{M}}_{14a} + \frac{t_n - \hat{t}_a}{\hat{t}_b - \hat{t}_a} \hat{\mathbf{M}}_{14b} \right) \begin{bmatrix} \hat{\mathbf{a}}_{d14n} \\ \hat{\mathbf{a}}_{c14n} \end{bmatrix}$$

3:   
$$\begin{bmatrix} \bar{\mathbf{a}}_{d25n} \\ \bar{\mathbf{a}}_{c25n} \end{bmatrix} = \left( \frac{\hat{t}_b - t_n}{\hat{t}_b - \hat{t}_a} \hat{\mathbf{M}}_{25a} + \frac{t_n - \hat{t}_a}{\hat{t}_b - \hat{t}_a} \hat{\mathbf{M}}_{25b} \right) \begin{bmatrix} \hat{\mathbf{a}}_{d25n} \\ \hat{\mathbf{a}}_{c25n} \end{bmatrix}$$

4:   
$$\begin{bmatrix} \bar{\mathbf{a}}_{d36n} \\ \bar{\mathbf{a}}_{c36n} \end{bmatrix} = \left( \frac{\hat{t}_b - t_n}{\hat{t}_b - \hat{t}_a} \hat{\mathbf{M}}_{36a} + \frac{t_n - \hat{t}_a}{\hat{t}_b - \hat{t}_a} \hat{\mathbf{M}}_{36b} \right) \begin{bmatrix} \hat{\mathbf{a}}_{d36n} \\ \hat{\mathbf{a}}_{c36n} \end{bmatrix}$$

5: end for

```

---

## 6.2 Gross outlier detection and removal

It is advisable to remove gross outliers prior to the angular rate reconstruction since their effects would be "smeared out" in the angular rate and acceleration reconstruction that involves numerical integration and filtering. We employ the following simple algorithm for detecting and removing gross outliers. The algorithm relies on the fact that moving-median filters are (a) robust against outliers, spikes, etc. (b) preserve edges and step functions, and (c) behave like low-pass filters. A symmetric moving median filter is defined by

$$y_n = \text{median}(x_{n-k}, x_{n-k+1}, \dots, x_{n+k}) \tag{70}$$

---

**Algorithm 9** Gradiometer calibration (science mode)
 

---

```

1: for  $n \leftarrow n_a, n_b$  do
2:    $\begin{bmatrix} \bar{\mathbf{a}}_{d14n} \\ \bar{\mathbf{a}}_{c14n} \end{bmatrix} = \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{M}}_{14a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{M}}_{14b} \right) \begin{bmatrix} \bar{\mathbf{a}}_{d14n} \\ \bar{\mathbf{a}}_{c14n} \end{bmatrix}$ 
3:    $\begin{bmatrix} \bar{\mathbf{a}}_{d25n} \\ \bar{\mathbf{a}}_{c25n} \end{bmatrix} = \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{M}}_{25a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{M}}_{25b} \right) \begin{bmatrix} \bar{\mathbf{a}}_{d25n} \\ \bar{\mathbf{a}}_{c25n} \end{bmatrix}$ 
4:    $\begin{bmatrix} \bar{\mathbf{a}}_{d36n} \\ \bar{\mathbf{a}}_{c36n} \end{bmatrix} = \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{M}}_{36a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{M}}_{36b} \right) \begin{bmatrix} \bar{\mathbf{a}}_{d36n} \\ \bar{\mathbf{a}}_{c36n} \end{bmatrix}$ 
5:    $\begin{bmatrix} \bar{\mathbf{a}}_{d14n} \\ \bar{\mathbf{a}}_{c14n} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{a}}_{d14n} \\ \bar{\mathbf{a}}_{c14n} \end{bmatrix} + \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{K}}_{14a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{K}}_{14b} \right) \begin{bmatrix} (\bar{\mathbf{a}}_{c14n} + \bar{\mathbf{a}}_{d14n})^2 \\ (\bar{\mathbf{a}}_{c14n} - \bar{\mathbf{a}}_{d14n})^2 \end{bmatrix}$ 
6:    $\begin{bmatrix} \bar{\mathbf{a}}_{d25n} \\ \bar{\mathbf{a}}_{c25n} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{a}}_{d25n} \\ \bar{\mathbf{a}}_{c25n} \end{bmatrix} + \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{K}}_{25a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{K}}_{25b} \right) \begin{bmatrix} (\bar{\mathbf{a}}_{c25n} + \bar{\mathbf{a}}_{d25n})^2 \\ (\bar{\mathbf{a}}_{c25n} - \bar{\mathbf{a}}_{d25n})^2 \end{bmatrix}$ 
7:    $\begin{bmatrix} \bar{\mathbf{a}}_{d36n} \\ \bar{\mathbf{a}}_{c36n} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{a}}_{d36n} \\ \bar{\mathbf{a}}_{c36n} \end{bmatrix} + \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{K}}_{36a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{K}}_{36b} \right) \begin{bmatrix} (\bar{\mathbf{a}}_{c36n} + \bar{\mathbf{a}}_{d36n})^2 \\ (\bar{\mathbf{a}}_{c36n} - \bar{\mathbf{a}}_{d36n})^2 \end{bmatrix}$ 
8:    $\begin{bmatrix} \bar{\mathbf{a}}_{d14n} \\ \bar{\mathbf{a}}_{c14n} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{a}}_{d14n} \\ \bar{\mathbf{a}}_{c14n} \end{bmatrix} + \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{W}}_{14a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{W}}_{14b} \right) \dot{\bar{\omega}}$ 
9:    $\begin{bmatrix} \bar{\mathbf{a}}_{d25n} \\ \bar{\mathbf{a}}_{c25n} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{a}}_{d25n} \\ \bar{\mathbf{a}}_{c25n} \end{bmatrix} + \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{W}}_{25a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{W}}_{25b} \right) \dot{\bar{\omega}}$ 
10:   $\begin{bmatrix} \bar{\mathbf{a}}_{d36n} \\ \bar{\mathbf{a}}_{c36n} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{a}}_{d36n} \\ \bar{\mathbf{a}}_{c36n} \end{bmatrix} + \left( \frac{\bar{t}_b - t_n}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{W}}_{36a} + \frac{t_n - \bar{t}_a}{\bar{t}_b - \bar{t}_a} \bar{\mathbf{W}}_{36b} \right) \dot{\bar{\omega}}$ 
11: end for

```

---

where  $x_n$  is the filter input,  $y_n$  is the filter output, and  $2k + 1$  is the width of the moving window. When subtracting the filter output from the filter input, i.e.

$$e_n = x_n - \text{median}(x_{n-k}, x_{n-k+1}, \dots, x_{n+k}), \quad (71)$$

the residuals  $e_n$  should contain mainly high-frequency noise and features like outliers, spikes, etc. When the width of the moving window is chosen appropriately, the outliers, spikes, etc. will not be "smeared out", as would be the case for a moving-average filter, which makes it easy to detect them.

An outlier is detected, if the absolute value of  $e_n$  exceeds the threshold  $k$ , i.e.

$$\text{abs}(e_n) > k. \quad (72)$$

The outliers detected in this way are marked as invalid epochs. In addition to these outliers, we mark  $M$  epochs before and after the outlier. The marked epochs in  $x_n$  are then replaced by linear interpolated values, where the last valid epoch before and after the outlier are used for interpolation.

In practice, we use the calibrated differential acceleration  $\bar{\mathbf{a}}_{dij}$  as input for detecting outliers. In case we find an outlier in one of the nine time series (three axes for each  $ij \in \{14, 25, 36\}$ ), we consider that all acceleration  $\bar{\mathbf{a}}_{dij}$  and  $\bar{\mathbf{a}}_{cij}$  are affected by outliers. At the beginning and the end of the time series, the window width of the moving-median filter is shortened such that the filter does not access the epochs  $n < 1$  or  $n > N$ , noting that  $n = 1$  is the first epoch and  $n = N$  is the last epoch, and the window is still centred around epoch  $n$ . This implies  $e_1 = e_N = 0$  by definition, which means that the algorithm will never detect an outlier in the first or last epoch.

---

**Algorithm 10** Gross outlier removal (part 1)
 

---

```

1: for  $n \leftarrow 1, N$  do
2:    $f_n = 1$  ▷ Initialise flags
3: end for
4: for  $ij \leftarrow 14, 25, 36$  do ▷ Loop over all differential acceleration
5:   for  $\alpha \leftarrow x, y, z$  do
6:     for  $n \leftarrow 1, N$  do ▷ Loop over all epochs
7:       if  $n \leq W$  then ▷ Use shorter filter
8:          $e_{dij\alpha, n} = \bar{a}_{dij\alpha, n} - \text{median}(\bar{a}_{dij\alpha, 1}, \bar{a}_{dij\alpha, 2}, \dots, \bar{a}_{dij\alpha, 2n-1})$ 
9:       else if  $n \geq N - W$  then ▷ Use shorter filter
10:         $e_{dij\alpha, n} = \bar{a}_{dij\alpha, n} - \text{median}(\bar{a}_{dij\alpha, 2n-N}, \bar{a}_{dij\alpha, 2n-N+1}, \dots, \bar{a}_{dij\alpha, N})$ 
11:       else
12:         $e_{dij\alpha, n} = \bar{a}_{dij\alpha, n} - \text{median}(\bar{a}_{dij\alpha, n-W}, \bar{a}_{dij\alpha, n-W+1}, \dots, \bar{a}_{dij\alpha, n+W})$ 
13:       end if
14:       if  $\text{abs}(e_{dij\alpha, n}) > k_{dij\alpha}$  then
15:         for  $m \leftarrow \max(n - M, 1), \min(n + M, N)$  do
16:            $f_m = 0$  ▷ Mark outlier by setting flag to zero
17:         end for
18:       end if
19:     end for
20:   end for
21: end for

```

---

---

**Algorithm 10** Gross outlier removal (part 2)
 

---

```

22:  $f_{first} = f_1$  ▷ Save first and last flag
23:  $f_{last} = f_N$ 
24:  $f_1 = 1$  ▷ Ensure that first and last epoch can be used in linear interpolation
25:  $f_N = 1$ 
26: for  $n \leftarrow 1, N$  do ▷ Search for flagged outliers
27:   if  $f_n = 0$  then
28:      $n_a = n - 1$  ▷ Index of last valid epoch before outlier
29:     for  $k \leftarrow n + 1, N$  do
30:       if  $f_k = 1$  then
31:          $n_b = k$  ▷ Index of first valid epoch after outlier
32:         break ▷ Interrupt the for-loop
33:       end if
34:     end for
35:     for  $ij \leftarrow 14, 25, 36$  do ▷ Loop over all differential acceleration
36:       for  $\alpha \leftarrow x, y, z$  do
37:         for  $k \leftarrow n_a + 1, n_b - 1$  do ▷ Linear interpolation of flagged value
38:            $\bar{a}_{dij\alpha,k} = \frac{t_b - t_k}{t_b - t_a} \bar{a}_{dij\alpha,n_a} + \frac{t_k - t_a}{t_b - t_a} \bar{a}_{dij\alpha,n_b}$ 
39:         end for
40:       end for
41:     end for
42:   end if
43: end for
44:  $f_1 = f_{first}$  ▷ Restore flag of first and last epoch
45:  $f_N = f_{last}$ 

```

---





Inputs	Symbol	Unit	Contents	Source algorithm
Epochs	$\bar{t}_a, \bar{t}_b$	$s$	References epochs in GPS seconds	Calibration against star tracker angular rates and gravity field model
Acceleration	$\bar{\mathbf{a}}_{d14}, \bar{\mathbf{a}}_{c14}, \bar{\mathbf{a}}_{d25}, \bar{\mathbf{a}}_{c25}, \bar{\mathbf{a}}_{d36}, \bar{\mathbf{a}}_{c36}$	$m/s^2$	Shaking mode calibrated common and differential mode acceleration	Shaking mode calibration
Angular acceleration	$\dot{\bar{\omega}}$	$rad/s^2$	Angular acceleration proxy	Angular acceleration reconstruction algorithm jadd refj
Inverse calibration matrices	$\bar{\mathbf{M}}_{14a}, \bar{\mathbf{M}}_{36a}, \bar{\mathbf{M}}_{25b}, \bar{\mathbf{M}}_{36b}, \bar{\mathbf{M}}_{25a}, \bar{\mathbf{M}}_{14b}$	unitless	Inverse calibration matrices at epochs $t_a$ and $t_b$	Calibration against star tracker angular rates and gravity field model
Quadratic factor matrices	$\bar{\mathbf{K}}_{14a}, \bar{\mathbf{K}}_{36a}, \bar{\mathbf{K}}_{25b}, \bar{\mathbf{K}}_{36b}, \bar{\mathbf{K}}_{25a}, \bar{\mathbf{K}}_{14b}$	$s^2/m$	Quadratic factor matrices at epochs $t_a$ and $t_b$	Calibration against star tracker angular rates and gravity field model
Angular acceleration coupling matrices	$\bar{\mathbf{W}}_{14a}, \bar{\mathbf{W}}_{36a}, \bar{\mathbf{W}}_{25b}, \bar{\mathbf{W}}_{36b}, \bar{\mathbf{W}}_{25a}, \bar{\mathbf{W}}_{14b}$	$m/rad$	Angular acceleration coupling matrices at epochs $t_a$ and $t_b$	Calibration against star tracker angular rates and gravity field model
Outputs	Symbol	Unit	Contents	
Acceleration	$\bar{\bar{\mathbf{a}}}_{d14}, \bar{\bar{\mathbf{a}}}_{c14}, \bar{\bar{\mathbf{a}}}_{d25}, \bar{\bar{\mathbf{a}}}_{c25}, \bar{\bar{\mathbf{a}}}_{d36}, \bar{\bar{\mathbf{a}}}_{c36}$	$m/s^2$	Calibrated common and differential mode acceleration	

Table 9: List of inputs and outputs of the gradiometer calibration algorithm (science mode)



Inputs	Symbol	Unit	Contents	Source algorithm
Acceleration	$\bar{\mathbf{a}}_{d14}, \bar{\mathbf{a}}_{c14}, \bar{\mathbf{a}}_{d25}, \bar{\mathbf{a}}_{c25}, \bar{\mathbf{a}}_{d36}, \bar{\mathbf{a}}_{c36}$	$m/s^2$	Calibrated common and differential mode acceleration	Gradiometer calibration
Detection threshold	$\mathbf{k}_{d14}, \mathbf{k}_{d25}, \mathbf{k}_{d36}$	$m/s^2$	Thresholds for detecting gross outliers	
Half window width	$W$	number of epochs	Half window width of moving-median filter	
Number of epochs	$M$	number of epochs	Number of epochs that will be flagged before and after detected outliers	
Outputs	Symbol	Unit	Contents	
Acceleration	$\bar{\mathbf{a}}_{d14}, \bar{\mathbf{a}}_{c14}, \bar{\mathbf{a}}_{d25}, \bar{\mathbf{a}}_{c25}, \bar{\mathbf{a}}_{d36}, \bar{\mathbf{a}}_{c36}$	$m/s^2$	Calibrated common and differential mode acceleration without gross outliers	
Flags	$f_n$	unitless	Flag indicating if data is interpolated (1 = not interpolated, 0 = interpolated)	

Table 10: List of inputs and outputs of the outlier removal algorithm



## 7 Angular rate and acceleration reconstruction

The star trackers provide the inertial attitude, from which we can determine the angular rates of the satellite by differentiation of the attitude quaternions. The differentiation tilts the star tracker noise PSD such that high-frequency is amplified and low-frequency noise is dampened. The star tracker angular rates are therefore accurate at low frequencies and less accurate at high frequencies. We can determine the angular rates also by integrating the gradiometer angular rates (except for the integration constant), where the integration tilts the gradiometer noise PSD such that low-frequency noise is amplified and high-frequency noise is dampened. Thus, the gradiometer angular rates are accurate at high frequencies and less accurate at low frequencies. Obviously, the star tracker and gradiometer angular rates are synergetic and the angular rate and acceleration reconstruction takes advantage of this fact. In a nutshell, we apply a lowpass filter to the star tracker angular rates and a complementary highpass filter to the gradiometer angular rates, and add results to arrive at the reconstructed angular rates. The reconstructed angular accelerations are obtained by differentiating the reconstructed angular rates.

In the following, we provide details on the calculation of the star tracker and gradiometer angular rates as well as the angular rate and acceleration reconstruction algorithm. We will keep the naming of the variables generic because the algorithm is used twice in the processing, one time prior to the gradiometer calibration to obtain a proxy for the angular accelerations of the satellite and another time after the calibration of the gradiometer and correction of misalignments between star trackers and gradiometer.

### 7.1 Calculation of star tracker angular rates

The calculation of angular rates from the combined star tracker quaternions is straight forward using Eqs. (34) and (35). Due to the sign ambiguity of quaternions, we need to run Algorithm 4 prior to the differentiation of quaternions. For the epochs when none of the star sensors is providing a valid attitude, we interpolate the quaternions in order to be able to calculate the angular rates in all cases. This is needed because the filtering applied in the angular rate reconstruction is not designed to handle data gaps. Even a single missing attitude quaternion would lead to a data gap of the length of the reconstruction filters, which is avoided by the interpolation. We use cubic spline interpolation as specified in Section 2.2. The calculation of angular rates is detailed in Algorithm 11.



Inputs	Symbol	Unit	Contents	Source algorithm
Attitude quaternions	$\mathbf{q}$	unitless	Orientation of either $CRF$ or $GRF$ wrt. IRF	Either star tracker combination or star tracker misalignment adjustment
Epochs	$t$	GPS second	Epochs of $\mathbf{q}$	Star tracker combination
Flags	$\mathbf{f}_q$	unitless	Flags for star tracker quaternions (1 = valid, 0 = invalid)	Star tracker combination
Time step	$\Delta t$	seconds	Time step for approximating the first time derivative (shall be much smaller than sampling rate)	Control parameter

Outputs	Symbol	Unit	Contents
Angular rates	$\boldsymbol{\omega}$	$rad/s$	Angular rates from combined star tracker quaternions
Flags	$\mathbf{f}_\omega$	unitless	Flags for angular rates (1 = valid, 0 = invalid)

Table 11: List of inputs and outputs of algorithm for calculation of star tracker angular rates

## 7.2 Calculation of gradiometer angular rates

The angular rates from the gradiometer are calculated by numerical integration of the gradiometer angular accelerations as described in Algorithm 1. These processing steps are summarised in Algorithm 12.

## 7.3 Calculation of filters for angular rate reconstruction

We use the same approach for calculating the angular rate reconstruction filters as Stummer et al. [2011], with the difference that we define the noise PSDs for star tracker and gradiometer angular rates differently. Here, we choose a simpler model that allows us to choose very easily the frequency where the PSDs cross. It is defined by

$$\mathbf{P}_S = \mathbf{f}^{\alpha_S} \quad \text{and} \quad \mathbf{P}_G = c \mathbf{f}^{\alpha_G} \tag{73}$$

where  $\mathbf{f}$  is the frequency vector,  $\mathbf{P}_S$  and  $\mathbf{P}_G$  are the PSD of the star tracker and gradiometer angular rates, respectively,  $\alpha_S$  and  $\alpha_G$  are the slope of these PSDs in the logarithmic domain, and

$$c = (f^{cross})^{\alpha_S - \alpha_G} \tag{74}$$

---

**Algorithm 11** Calculation of star tracker angular rates
 

---

```

1: Run Algorithm 4 on the star tracker quaternions  $\mathbf{q}$  to make them continuous
2: Interpolate all quaternions for which  $f_{q,n} = 0$  using cubic spline interpolation as specified
   in Section 2.2. Flags of interpolated quaternions remain  $f_{q,n} = 0$ .
3: for  $n \leftarrow 1, N$  do                                     ▷ Normalize all quaternions (including interpolated ones)
4:    $q_n = q_n / \sqrt{q_n^T q_n}$ 
5: end for
6:  $\dot{\mathbf{q}} = \text{differentiate}(\mathbf{t}, \mathbf{q}, \Delta t)$                                                          ▷ Algorithm 2
7:  $W_1 = q_1^* \dot{q}_1$                                                                                    ▷ Quaternion multiplication
8:  $\boldsymbol{\omega}_1 = 2 \begin{bmatrix} W_{1,2} & W_{1,3} & W_{1,4} \end{bmatrix}^T$ 
9:  $f_{\omega,1} = f_{q,1} f_{q,2}$                                                                                    ▷ Multiply flags
10: for  $n \leftarrow 2, N - 1$  do
11:    $W_n = q_n^* \dot{q}_n$                                                                                    ▷ Quaternion multiplication
12:    $\boldsymbol{\omega}_n = 2 \begin{bmatrix} W_{n,2} & W_{n,3} & W_{n,4} \end{bmatrix}^T$ 
13:    $f_{\omega,n} = f_{q,n-1} f_{q,n} f_{q,n+1}$                                                                  ▷ Multiply flags
14: end for
15:  $W_N = q_N^* \dot{q}_N$                                                                                    ▷ Quaternion multiplication
16:  $\boldsymbol{\omega}_N = 2 \begin{bmatrix} W_{N,2} & W_{N,3} & W_{N,4} \end{bmatrix}^T$ 
17:  $f_{\omega,N} = f_{q,N-1} f_{q,N}$                                                                                    ▷ Multiply flags

```

---



---

**Algorithm 12** Calculation of gradiometer angular rates
 

---

```

1:  $\dot{\boldsymbol{\omega}}_x = -\mathbf{a}_{d36y} / L_z + \mathbf{a}_{d25z} / L_y$ 
2:  $\dot{\boldsymbol{\omega}}_y = -\mathbf{a}_{d14z} / L_x + \mathbf{a}_{d36x} / L_z$ 
3:  $\dot{\boldsymbol{\omega}}_z = -\mathbf{a}_{d25x} / L_y + \mathbf{a}_{d14y} / L_x$ 
4:  $\boldsymbol{\omega}_x = \text{integrate}(\mathbf{t}, \dot{\boldsymbol{\omega}}_x, K)$                                                                  ▷ Algorithm 1
5:  $\boldsymbol{\omega}_y = \text{integrate}(\mathbf{t}, \dot{\boldsymbol{\omega}}_y, K)$ 
6:  $\boldsymbol{\omega}_z = \text{integrate}(\mathbf{t}, \dot{\boldsymbol{\omega}}_z, K)$ 

```

---

is a scale factor depending on the frequency  $f^{cross}$  that defines where  $\mathbf{P}_S$  and  $\mathbf{P}_G$  cross each other. The length of the frequency vector is equal to the length of the filters, which we denote by  $N_F$  and must be an odd integer that is large enough to achieve sufficient resolution in the spectral domain. We recommend to use

$$N_F \approx \frac{10}{f^{cross}}. \quad (75)$$

Since the components  $x$ ,  $y$  and  $z$  of the angular rates are reconstructed independently, we omit the subscripts  $x$ ,  $y$  and  $z$  in the following for simplicity. In practice, we have to run the algorithms for the calculation of the angular rate reconstruction filters as well as the angular rate reconstruction itself three times, i.e. once per component. It is possible to use different input parameters such as  $f^{cross}$  for each component.



Inputs	Symbol	Unit	Contents	Source algorithm
Accelerations	$\mathbf{a}_{dij}$	$m/s^2$	Calibrated accelerations	Either satellite shaking or science mode gradiometer calibration
Epochs	$t$	GPS second	Gradiometer measurement epochs	EGG_NOM_1B files
Gradiometer arm length	$L_x, L_y, L_z$	meters	Gradiometer arm length	
Factor	$K$	unitless	Factor defining the increase of epochs of the upsampled time series	Control parameter
Outputs	Symbol	Unit	Contents	
Angular rates	$\boldsymbol{\omega}$	$rad/s$	Angular rates from gradiometer	

Table 12: List of inputs and outputs of algorithm for calculation of gradiometer angular rates

Inputs	Symbol	Unit	Contents
Crossing frequency	$f^{cross}$	Hz	Frequency of equal spectral weights
Length of filter	$N_F$	unitless	Length of filter (odd integer)
Exponent	$\alpha_S, \alpha_G$	unitless	Slopes of PSDs in logarithmic domain
Outputs	Symbol	Unit	Contents
Filter coefficients	$\mathbf{F}_S, \mathbf{F}_G$	unitless	Filter coefficients for one component ( $x, y$ or $z$ ) of the angular rates

Table 13: List of inputs and outputs for calculation of angular rate reconstruction filters

## 7.4 Application of filters for angular rate reconstruction

The angular rate reconstruction filters calculated according to Algorithm 13 are symmetric moving-average filters. Applying the filters is therefore a convolution in the time domain, which can be efficiently performed as an element-wise multiplication in the frequency domain. We use the symbol  $\odot$  for notating elementwise multiplication, i.e.

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} \odot \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} = \begin{bmatrix} a_1 b_1 \\ a_2 b_2 \\ \vdots \\ a_N b_N \end{bmatrix}. \tag{76}$$

---

**Algorithm 13** Calculation of filters for angular rate reconstruction
 

---

```

1:  $K = \text{ceil}(N_F/2)$  ▷ Index of mid frequency
2:  $\mathbf{f} = \frac{1}{N_F} [0 \ 1 \ \dots \ N_F - 1]^T$  ▷ Frequency vector
3:  $\mathbf{c} = (\mathbf{f}^{\text{cross}})^{\alpha_S - \alpha_G}$ 
4:  $\mathbf{P}_G = \mathbf{c} \mathbf{f}^{\alpha_G}$  ▷ Define PSDs
5:  $\mathbf{P}_S = \mathbf{f}^{\alpha_S}$ 
6: for  $n \leftarrow 2, K$  do ▷ Make PSD symmetric around mid frequency
7:    $P_{S, N_F - n + 2} = P_{S, n}$ 
8:    $P_{G, N_F - n + 2} = P_{G, n}$ 
9: end for
10:  $\mathbf{W}_S = \text{zeros}(\text{size}(\mathbf{P}_S))$  ▷ Spectral weights for star tracker angular rates
11: if  $\alpha_S - \alpha_G < 0$  then ▷ Spectral weight for zero frequency
12:    $W_{S,1} = 0$ 
13: else if  $\alpha_S - \alpha_G > 0$  then
14:    $W_{S,1} = 1$ 
15: else
16:    $W_{S,1} = c/(c + 1)$ 
17: end if
18: for  $n \leftarrow 2, N_F$  do
19:    $W_{S,n} = P_{G,n}/(P_{G,n} + P_{S,n})$  ▷ Spectral weight for non-zero frequencies
20: end for
21:  $\mathbf{F}_S = \text{ifft}(\mathbf{W}_S)$  ▷ Filter coefficients
22:  $\mathbf{F}_S = [F_{S,K+1} \ F_{S,K+2} \ \dots \ F_{S,N_F} \ F_{S,1} \ F_{S,2} \ \dots \ F_{S,K}]$  ▷ Resort coefficients
23:  $\mathbf{F}_G = -\mathbf{F}_S$  ▷ Complementary filter
24:  $F_{G,K} = F_{G,K} + 1$ 

```

---

Generally, symmetric moving-average filters produce transient effects at the beginning and the end of the angular rate time series. Instead of cropping the filtered time series by half the filter length, we use a different approach for reducing the transient effects. Since the length  $N_F$  of the filters is an input parameter to Algorithm 13, it is straight forward to generate shorter filters. For the first epoch of the filtered time series, we create a filter of length  $N_F = 1$  and apply it to the first value of the input time series; for the second epoch of the filtered time series, we create a filter of length  $N_F = 3$  and apply it to the first three values of the input time series; and so forth until we have reached half the length of the filter. We repeat the same procedure for the end of the filtered time series. This approach avoids transient effects except for the first and last few epochs of the filtered time series, for which we extrapolate the gradiometer angular rates after fitting them to the sum of the filtered star tracker and gradiometer angular rates. The entire approach is detailed in Algorithm 14.



Inputs	Symbol	Unit	Contents
Crossing frequency	$f^{cross}$	Hz	Frequency of equal spectral weights
Length of filter	$N_F$	unitless	Length of filter (odd integer)
Exponent	$\alpha_S, \alpha_G$	unitless	Slopes of PSDs in logarithmic domain
Number of epochs	$M$	unitless	Number of epochs at beginning/end
Angular rates	$\omega_S, \omega_G$	rad/s	One component ( $x, y$ or $z$ ) of the star trackers and gradiometer angular rates
Outputs	Symbol	Unit	Contents
Angular rates	$\omega$	rad/s	One component ( $x, y$ or $z$ ) of the reconstructed angular rates
Angular accelerations	$\dot{\omega}$	rad/s <sup>2</sup>	One component ( $x, y$ or $z$ ) of the reconstructed angular accelerations (optional)

Table 14: List of inputs and outputs of angular rate reconstruction algorithm

---

**Algorithm 14** Angular rate reconstruction (part 1)

---

- 1:  $K = \text{floor}(N_F/2)$
  - 2: Create  $\mathbf{F}_G$  and  $\mathbf{F}_S$  of length  $N_F$  using Algorithm 13
  - 3:  $N_\omega = \text{length}(\omega_G)$  ▷ Same length as  $\omega_S$
  - 4:  $N_{FFT} = 2^{\text{ceil}(\log_2(N_F+N_\omega-1))}$  ▷ Convolution of filters and angular rates
  - 5:  $\mathbf{h} = \text{ifft}(\text{fft}(\mathbf{F}_G, N_{FFT}) \odot \text{fft}(\omega_G, N_{FFT}) + \text{fft}(\mathbf{F}_S, N_{FFT}) \odot \text{fft}(\omega_S, N_{FFT}))$
  - 6:  $\omega = [h_{K+1} \ h_{K+2} \ \dots \ h_{K+N_\omega}]^T$
  - 7: **for**  $n \leftarrow 1, \min(K, \text{ceil}(N_\omega/2))$  **do** ▷ Apply shorter filters at beginning and end
  - 8:     Create  $\mathbf{F}_G$  and  $\mathbf{F}_S$  of length  $2n - 1$  using Algorithm 13
  - 9:      $\omega_n = \mathbf{F}_G^T \begin{bmatrix} \omega_{G,1} \\ \omega_{G,2} \\ \vdots \\ \omega_{G,2n-1} \end{bmatrix} + \mathbf{F}_S^T \begin{bmatrix} \omega_{S,1} \\ \omega_{S,2} \\ \vdots \\ \omega_{S,2n-1} \end{bmatrix}$
  - 10:      $\omega_{N_\omega-n+1} = \mathbf{F}_G^T \begin{bmatrix} \omega_{G,N_\omega-2n+2} \\ \omega_{G,N_\omega-2n+3} \\ \vdots \\ \omega_{G,N_\omega} \end{bmatrix} + \mathbf{F}_S^T \begin{bmatrix} \omega_{S,N_\omega-2n+2} \\ \omega_{S,N_\omega-2n+3} \\ \vdots \\ \omega_{S,N_\omega} \end{bmatrix}$
  - 11: **end for**
-



**Algorithm 14** Angular rate reconstruction (part 2)

12:  $\boldsymbol{\tau} = \frac{1}{2M} \begin{bmatrix} 1 & 2 & \dots & 2M \end{bmatrix}^T$   $\triangleright$  Replace first/last  $M$  elements of  $\boldsymbol{\omega}$  with trend-corrected  $\boldsymbol{\omega}_G$

13:  $\mathbf{A}_1 = \begin{bmatrix} \tau_1 & 1 - \tau_1 \\ \tau_2 & 1 - \tau_2 \\ \vdots & \vdots \\ \tau_{2M} & 1 - \tau_{2M} \end{bmatrix}$

14:  $\mathbf{A}_2 = \begin{bmatrix} \tau_{M+1} & 1 - \tau_{M+1} \\ \tau_{M+2} & 1 - \tau_{M+2} \\ \vdots & \vdots \\ \tau_{2M} & 1 - \tau_{2M} \end{bmatrix}$

15:  $\mathbf{p} = \frac{1}{2} + \frac{1}{2} \cos(\pi\boldsymbol{\tau})$

16:  $\Delta\boldsymbol{\omega} = \begin{bmatrix} \omega_{M+1} & \omega_{M+2} & \dots & \omega_{2M} \end{bmatrix}^T - \begin{bmatrix} \omega_{G,M+1} & \omega_{G,M+2} & \dots & \omega_{G,2M} \end{bmatrix}^T$

17:  $\mathbf{x} = (\mathbf{A}_2^T \mathbf{A}_2)^{-1} \mathbf{A}_2^T \Delta\boldsymbol{\omega}$

18:  $\begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_{2M} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} - \mathbf{p} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_{2M} \end{bmatrix} + \mathbf{p} \begin{bmatrix} \omega_{G,1} \\ \omega_{G,2} \\ \vdots \\ \omega_{G,2M} \end{bmatrix} + \mathbf{A}_1 \mathbf{x}$

19:  $\Delta\boldsymbol{\omega} = \begin{bmatrix} \omega_{N_\omega-M} & \omega_{N_\omega-M-1} & \dots & \omega_{N_\omega-2M+1} \end{bmatrix}^T - \begin{bmatrix} \omega_{G,N_\omega-M} & \omega_{G,N_\omega-M-1} & \dots & \omega_{G,N_\omega-2M+1} \end{bmatrix}^T$

20:  $\mathbf{x} = (\mathbf{A}_2^T \mathbf{A}_2)^{-1} \mathbf{A}_2^T \Delta\boldsymbol{\omega}$

21:  $\begin{bmatrix} \omega_{N_\omega} \\ \omega_{N_\omega-1} \\ \vdots \\ \omega_{N_\omega-2M+1} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} - \mathbf{p} \begin{bmatrix} \omega_{N_\omega} \\ \omega_{N_\omega-1} \\ \vdots \\ \omega_{N_\omega-2M+1} \end{bmatrix} + \mathbf{p} \begin{bmatrix} \omega_{G,N_\omega} \\ \omega_{G,N_\omega-1} \\ \vdots \\ \omega_{G,N_\omega-2M+1} \end{bmatrix} + \mathbf{A}_1 \mathbf{x}$

22:  $\dot{\boldsymbol{\omega}} = \text{differentiate}(\boldsymbol{\omega})$   $\triangleright$  optional, Algorithm 2



## 8 Attitude reconstruction

The goal of the attitude reconstruction is improving the attitude provided by the combined star sensors through incorporating the reconstructed angular rates. Since the latter describe the rotation from one epoch to the next, we fit the integrated reconstructed angular rates to a sequence of combined star tracker attitude quaternions. The principle is illustrated in Fig. 2. The attitude quaternions resulting purely from the reconstructed angular rates are expected to be much more smooth than the combined star tracker attitude quaternions, because they include the gradiometer angular rates, which are more accurate than star tracker angular rates at high frequencies. However, small errors will accumulate in the integration of the reconstructed angular rates, so that the resulting integrated attitude quaternions are expected to be less accurate at low frequencies.

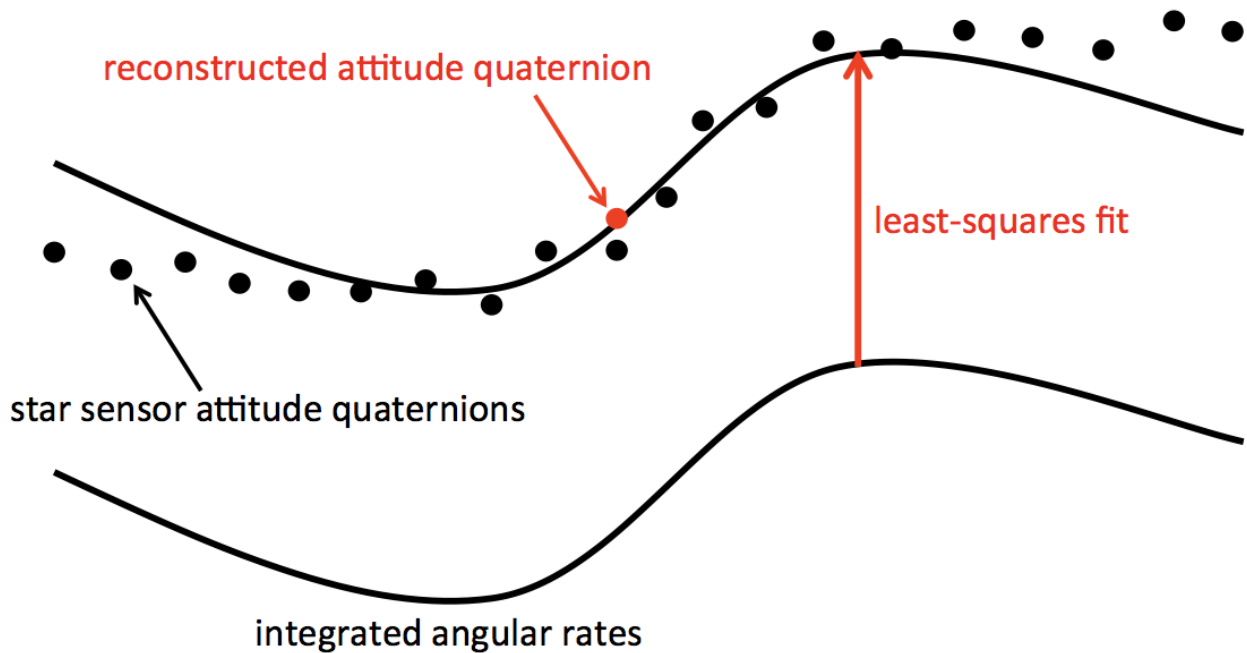


Figure 2: Principle of the proposed attitude reconstruction algorithm.

### 8.1 Mathematical derivation of algorithm

Before deriving the algorithm, it is instructive to introduce a different notation of variables. We denote the quaternions of the combined star trackers, which are corrected for the misalignment between star trackers and gradiometers, by  $q_n^{meas}$ , i.e.

$$q_n^{meas} = q_{GRF,n}^{IRF}. \tag{77}$$

Further, we define the quaternion  $q_n^{true}$  as the true (free of errors and noise) counterpart of  $q_n^{meas}$  and the rotation quaternion  $q_{n \rightarrow n+k}^{rot}$ , which describes the rotation of the gradiometer from epoch  $t_n$  to  $t_{n+k}$  in the sense of

$$q_{n \rightarrow n+k}^{rot} = (q_{GRF,n}^{IRF})^* q_{GRF,n+k}^{IRF}, \quad (78)$$

where it is emphasised that  $q_{n \rightarrow n+k}^{rot}$  is calculated by integrating the reconstructed angular rates, which we denote by  $\omega_n$ . With this notation, we can express the measured quaternion  $q_{n+k}^{meas}$  as a rotated version of the true quaternion  $q_n^{meas}$ , i.e.

$$q_{n+k}^{meas} = q_{n+k}^{noise} q_n^{true} q_{n \rightarrow n+k}^{rot} \quad (79)$$

where  $q_{n+k}^{noise}$  reflects the combined effect of the noise of the combined attitude quaternion  $q_{n+k}^{meas}$  and the noise of the rotation quaternion  $q_{n \rightarrow n+k}^{rot}$  that results from the integration of the reconstructed angular rates. The noise quaternion  $q_{n+k}^{noise}$  is assumed to represent a small rotation, i.e.

$$q_{n+k}^{noise} = \begin{bmatrix} 1 \\ \epsilon_{x,n+k}/2 \\ \epsilon_{y,n+k}/2 \\ \epsilon_{z,n+k}/2 \end{bmatrix} \quad (80)$$

where  $\epsilon_{x,n+k}$ ,  $\epsilon_{y,n+k}$  and  $\epsilon_{z,n+k}$  are small angles as described in Section 3.

In order to explain the relation between the rotation quaternion  $q_{n \rightarrow n+k}^{rot}$  and the reconstructed angular rates  $\omega_n$ , we decompose  $q_{n \rightarrow n+k}^{rot}$  into a series of rotations,

$$q_{n \rightarrow n+k}^{rot} = q_{n \rightarrow n+1}^{rot} q_{n+1 \rightarrow n+2}^{rot} \cdots q_{n+k-1 \rightarrow n+k}^{rot}, \quad (81)$$

noting that the sequence of rotation is important as quaternions are not commutative. Each quaternion  $q_{n+i \rightarrow n+i+1}^{rot}$  where  $i \in \{0, \dots, k-1\}$  is calculated from the reconstructed angular rates according to

$$q_{n+i \rightarrow n+i+1}^{rot} = \begin{bmatrix} \cos(\phi/2) \\ \sin(\phi/2)\mathbf{e} \end{bmatrix} \quad (82)$$

where

$$\phi = \frac{|\omega_{n+i+1} + \omega_{n+i}|}{2} \quad (83)$$

is the angle of rotation and

$$\mathbf{e} = \frac{\omega_{n+i+1} + \omega_{n+i}}{|\omega_{n+i+1} + \omega_{n+i}|} \quad (84)$$

is the rotation axis.

The measured quaternion  $q_{n+k}^{meas}$  in Eq. (79) composes of the product of the noise quaternion  $q_{n+k}^{noise}$  and the true quaternion  $q_n^{true}$ , which are both unknown. Since this would lead us to the least-squares adjustment according to the non-linear mixed model (Gauß-Helmert model),

we reformulate Eq. (79) in the following such that we can use linear generalised least squares (Gauß-Markov model), which is much simpler to solve. We start with rearranging Eq. (79) to

$$q_{n+k}^{meas} (q_n^{true} q_{n \rightarrow n+k}^{rot})^* = q_{n+k}^{noise}. \quad (85)$$

Now we define the relative error of the quaternions of epochs  $n$  and  $n+k$  as

$$q_{n,n+k}^{rel} = q_n^{noise} (q_{n+k}^{noise})^*, \quad (86)$$

which we can calculate according to Eq. (85) as

$$\begin{aligned} q_{n,n+k}^{rel} &= q_n^{meas} (q_n^{true} q_{n \rightarrow n+k}^{rot})^* (q_{n+k}^{meas} (q_n^{true} q_{n \rightarrow n+k}^{rot})^*)^* \\ &= q_n^{meas} (q_n^{true})^* q_{n \rightarrow n+k}^{rot} (q_{n+k}^{meas})^* \\ &= q_n^{meas} q_{n \rightarrow n+k}^{rot} (q_{n+k}^{meas})^* \end{aligned} \quad (87)$$

noting that  $q_{n \rightarrow n}^{rot}$  is a unit quaternion, i.e.

$$q_{n \rightarrow n}^{rot} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \quad (88)$$

It is important to realise that we can calculate the relative error of the quaternions  $q_{n,n+k}^{rel}$  from Eq. (87), even though it is the product of two unknown noise quaternions as described in Eq. (86). Because of the latter, we assume that it represents a small rotation denoted by

$$q_{n,n+k}^{rel} = \begin{bmatrix} 1 \\ \delta_{x,n,n+k}/2 \\ \delta_{y,n,n+k}/2 \\ \delta_{z,n,n+k}/2 \end{bmatrix}. \quad (89)$$

By rearranging Eq. (86) to

$$q_{n,n+k}^{rel} q_{n+k}^{noise} = q_n^{noise} \quad (90)$$

we can express the noise of epoch  $n+k$  as the noise of epoch  $n$  multiplied by the relative error, or equivalently

$$\mathbf{d}_{n,n+k} + \mathbf{e}_{n+k} = \mathbf{e}_n + O(\mathbf{e}^2) \quad (91)$$

where

$$\mathbf{d}_{n,n+k} = \begin{bmatrix} \delta_{x,n,n+k} \\ \delta_{y,n,n+k} \\ \delta_{z,n,n+k} \end{bmatrix}, \quad \mathbf{e}_n = \begin{bmatrix} \epsilon_{x,n} \\ \epsilon_{y,n} \\ \epsilon_{z,n} \end{bmatrix} \quad \text{and} \quad \mathbf{e}_{n+k} = \begin{bmatrix} \epsilon_{x,n+k} \\ \epsilon_{y,n+k} \\ \epsilon_{z,n+k} \end{bmatrix}. \quad (92)$$



We can write the observation equations of the generalised least-squares adjustment based on Eq. (91) as

$$\underbrace{\begin{bmatrix} \mathbf{d}_{n,n-K} \\ \vdots \\ \mathbf{d}_{n,n-1} \\ \mathbf{0} \\ \mathbf{d}_{n,n+1} \\ \vdots \\ \mathbf{d}_{n,n+K} \end{bmatrix}}_{\mathbf{y}} + \underbrace{\begin{bmatrix} \mathbf{e}_{n-K} \\ \vdots \\ \mathbf{e}_{n-1} \\ \mathbf{e}_n \\ \mathbf{e}_{n+1} \\ \vdots \\ \mathbf{e}_{n+K} \end{bmatrix}}_{\mathbf{v}} = \underbrace{\begin{bmatrix} \mathbf{I} \\ \vdots \\ \mathbf{I} \\ \mathbf{I} \\ \mathbf{I} \\ \vdots \\ \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \tilde{\mathbf{e}}_n \tag{93}$$

where  $\mathbf{y}$  is the observation vector,  $\mathbf{v}$  is the residual vector,  $\mathbf{A}$  is the design matrix and  $\tilde{\mathbf{e}}_n$  are the parameters.

## 8.2 Covariance matrix

Before we can calculate the parameters  $\tilde{\mathbf{e}}_n$  in Eq. (93) according to generalised least-squares, we need to construct the covariance matrix. For that purpose we need to consider the covariance matrices of  $\mathbf{e}_{n+k}$  for  $k = -K, \dots, K$  and their correlations. As mentioned before,  $q_{n+k}^{noise}$ , and thus  $\mathbf{e}_{n+k}$ , reflects the noise from the combined star tracker attitude quaternions as well as reconstructed angular rates.

The star tracker attitude combination algorithm provides for each epoch the covariance information of the combined attitude quaternion. It depends on which star trackers are combined and it show large differences between the case when only one star tracker is available and the case when more than one star tracker is available. Since the stars that are in the field of view of the star tracker change slowly along the orbit, we expect that the combined star tracker quaternions show some time correlation between epochs that differ by a few minutes as well as epochs from one orbit to the next since the attitude almost repeats from one orbit to the next. Such correlations are most likely due to small, systematic attitude errors that are clearly visible when inspecting the inter-boresight angles.

The noise in the reconstructed angular rates affects the rotation quaternions  $q_{n \rightarrow n+k}^{rot}$ . It is obvious from Eq. (81) that the variance of the noise in  $q_{n \rightarrow n+k}^{rot}$  grows proportionally with  $k^2$  and also shows large time correlations due to the integration of the angular rates described by that equation. Furthermore, we expect correlations between the rotation quaternions and the star tracker quaternions because the latter were used in the calculation of the first.

We conclude from the discussion that the covariance matrix should in principle be fully populated. However, using a fully populated covariance matrix would result in an extremely high

computational effort since one would have to invert a large matrix for each epoch. Therefore, we ignore time correlations and reduce thereby the covariance matrix to a block-diagonal structure, with one  $3 \times 3$  block per epoch. The computational effort will still be high, but manageable for a single CPU. There are two properties that we take into account when constructing the covariance matrix. The first is the covariance information that we get from the star tracker combination and the second is the linearly increasing standard deviation of the noise in the rotation quaternions. The "slope" of the increase can be determined empirically from residuals between combined star tracker attitude quaternions, which are corrected for the misalignment between star trackers and quaternions, and the rotation quaternions resulting from the integration of the reconstructed angular rates.

The covariance matrix that we utilise has therefore a block-diagonal structure,

$$\Sigma = \begin{bmatrix} \Sigma_{n-K} & 0 & \cdots & 0 \\ 0 & \Sigma_{n-K+1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \Sigma_{n-K} \end{bmatrix}, \quad (94)$$

where each block of dimension  $3 \times 3$  is the sum

$$\Sigma_{n+k} = \Sigma_{n+k}^{meas} + \Sigma_{n+k}^{rot} \quad (95)$$

of the covariance matrix  $\Sigma_{n+k}^{meas}$  of the combined star sensor quaternion and the covariance matrix  $\Sigma_{n+k}^{rot}$  of the rotation quaternion.

The covariance matrices  $\Sigma_{n+k}^{meas}$  of the combined star sensor quaternion are an output of Algorithm 6, which provides the cofactor matrices  $\mathbf{Q}_1, \dots, \mathbf{Q}_{123}$  and the square-sum of residuals  $\Omega$ . The latter still needs to be divided by the redundancy of the least-squares adjustment, in which the combined star tracker quaternions were estimated, in order to obtain the a posteriori variance factor  $\sigma_0^2$ . The redundancy  $R$  can be calculated from the flags  $\mathbf{f}_{S_1}$ ,  $\mathbf{f}_{S_2}$  and  $\mathbf{f}_{S_3}$  of the resampled and star tracker quaternions and the flags  $\mathbf{f}$  of the combined star trackers by

$$R = 3 \sum_{n=1}^N (f_n - f_{S_1,n} - f_{S_2,n} - f_{S_3,n}), \quad (96)$$

which gives then an a posteriori variance factor

$$\sigma_0^2 = \frac{\Omega}{R}. \quad (97)$$

The covariance matrices for the different combinations of available star trackers are the product of the variance factor and the cofactor matrices, i.e.

$$\Sigma_{S_1,n+k}^{meas} = \sigma_0^2 \mathbf{Q}_1, \quad \Sigma_{S_2,n+k}^{meas} = \sigma_0^2 \mathbf{Q}_2, \quad \text{etc.} \quad (98)$$



Since the cofactor matrices are constant, we list their values in Eqs. (99–105). The a posteriori variance factor  $\hat{\sigma}_0^2$  is approximately  $\hat{\sigma}_0^2 = 13^2 \mu\text{rad}^2$ .

$$\mathbf{Q}_1 = \begin{bmatrix} 1.000121521459708 & 0.095222836108324 & -0.054435478192699 \\ 0.095222836108324 & 75.615532876845165 & -42.655022458413313 \\ -0.054435478192699 & -42.655022458413320 & 25.384345601751161 \end{bmatrix} \quad (99)$$

$$\mathbf{Q}_2 = \begin{bmatrix} 1.001561466393628 & -0.131503870039127 & 0.370525932944455 \\ -0.131503870039127 & 12.075017626225865 & -31.205022613483280 \\ 0.370525932944455 & -31.205022613483280 & 88.923420907434434 \end{bmatrix} \quad (100)$$

$$\mathbf{Q}_3 = \begin{bmatrix} 41.413359274606734 & 42.610626719061884 & -23.495051626509660 \\ 42.610626719061884 & 45.927359219359751 & -24.772473572404802 \\ -23.495051626509660 & -24.772473572404802 & 14.659281505935640 \end{bmatrix} \quad (101)$$

$$\mathbf{Q}_{12} = \begin{bmatrix} 0.500011447421263 & -0.002903273148382 & 0.004247636496021 \\ -0.002903273148382 & 1.919411345174103 & -1.616662280334516 \\ 0.004247636496021 & -1.616662280334516 & 2.502024213487087 \end{bmatrix} \quad (102)$$

$$\mathbf{Q}_{13} = \begin{bmatrix} 0.965936848197282 & 0.968247299647752 & -0.543564953526007 \\ 0.968247299647752 & 2.879509027619699 & -1.339618855085098 \\ -0.543564953526007 & -1.339618855085097 & 1.254213201185986 \end{bmatrix} \quad (103)$$

$$\mathbf{Q}_{23} = \begin{bmatrix} 0.790153006185965 & 0.391922949490108 & -0.408574304412965 \\ 0.391922949490108 & 1.085989398656493 & -0.709773601283014 \\ -0.408574304412965 & -0.709773601283014 & 1.515784721963927 \end{bmatrix} \quad (104)$$

$$\mathbf{Q}_{123} = \begin{bmatrix} 0.436398899448459 & 0.214237386367438 & -0.179540976548104 \\ 0.214237386367438 & 0.986366903459979 & -0.587553396581192 \\ -0.179540976548104 & -0.587553396581192 & 0.931510054952443 \end{bmatrix} \quad (105)$$

The covariance matrix of the rotation quaternion  $\Sigma_{n+k}^{rot}$  is modelled as the diagonal matrix

$$\Sigma_{n+k}^{rot} = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_z^2 \end{bmatrix} \quad (106)$$



where the variances  $\sigma_x^2$ ,  $\sigma_y^2$  and  $\sigma_z^2$  are a quadratic functions of the time difference, i.e. they depend only on the index  $k$ . The functions are determined empirically from the data. Fig. 3 shows the residuals between reconstructed attitude quaternions and the quaternions that result from rotating an initial quaternion. This was performed for a number of epochs in order to be able to calculate the variance. We can clearly see that the errors increase quadratically with the time difference.

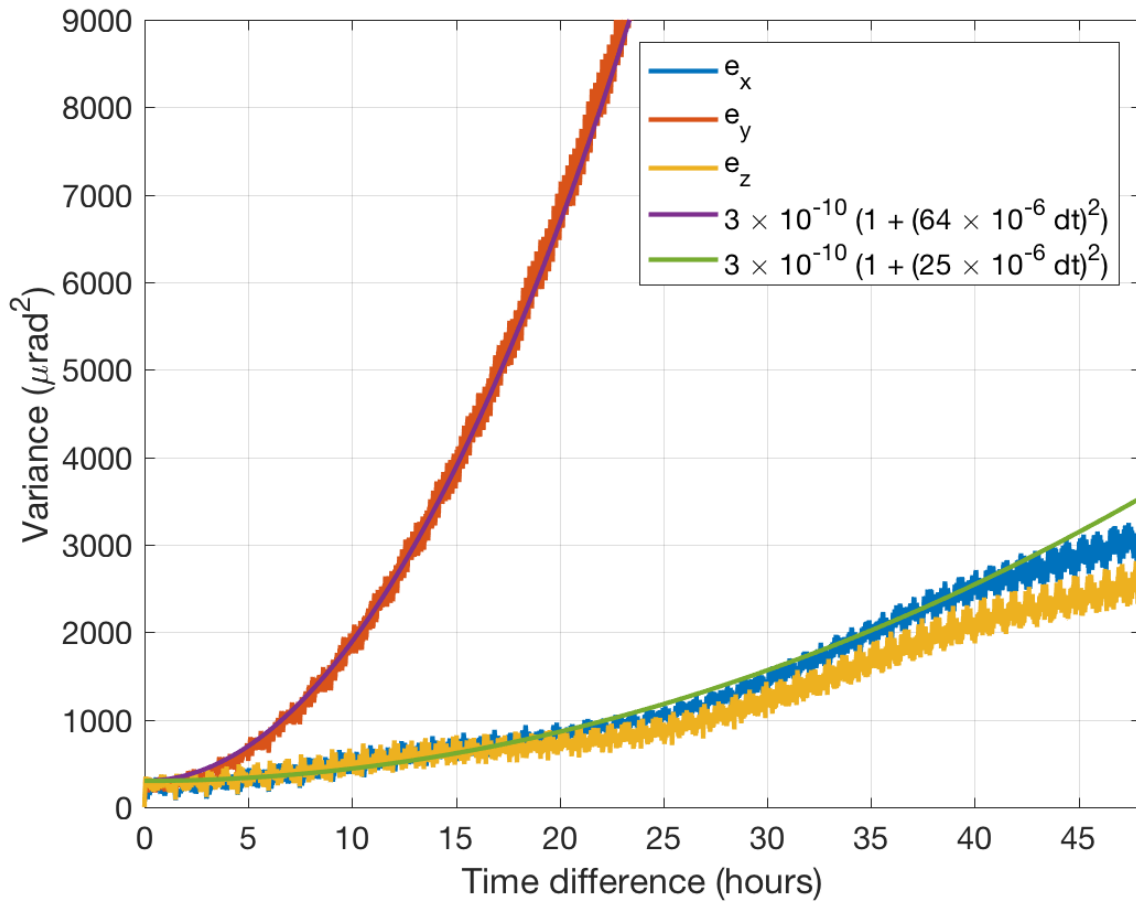


Figure 3: Variance between rotation quaternions and reconstructed attitude.

Figure 3 also shows two functions that approximate the increasing variance of the residuals. We use these functions to model the variances

$$\sigma_x^2 = 3 \times 10^{-10} (25 \times 10^{-6} \times \Delta t_k)^2 = \sigma_{0,x}^2 \Delta t_k^2, \tag{107}$$

$$\sigma_y^2 = 3 \times 10^{-10} (64 \times 10^{-6} \times \Delta t_k)^2 = \sigma_{0,y}^2 \Delta t_k^2 \tag{108}$$

and

$$\sigma_z^2 = 3 \times 10^{-10} (25 \times 10^{-6} \times \Delta t_k)^2 = \sigma_{0,z}^2 \Delta t_k^2 \tag{109}$$

where  $\Delta t_k = |t_{n+k} - t_n|$ . Note that the purpose of the constant  $3 \times 10^{-10} \text{rad}^2$  in Fig. 3 is accounting for the variance of the combined attitude quaternion.



### 8.3 Algorithm

The attitude reconstruction is detailed in Algorithm 15. It should be noted that in case some combined star tracker quaternions are not available (flag is zero), they will simply not be used in the attitude reconstruction, with the only exception of the centre epoch for which we reconstruct the attitude quaternion. Technically, they receive zero weight in the least squares estimation by setting

$$\Sigma_{n+k}^{-1} = (\Sigma_{n+k}^{meas} + \Sigma_{n+k}^{rot})^{-1} = 0. \tag{110}$$

Inputs	Symbol	Unit	Contents	Source algorithm
Attitude quaternions	$\mathbf{q}^{meas}$	unitless	Orientation of GRF wrt. IRF	Star tracker combination
Angular rates	$\boldsymbol{\omega}$	rad/s	Rotation rate of GRF wrt. IRF	Angular rate reconstruction
Flag for star trackers	$\mathbf{f}_{S_1}, \mathbf{f}_{S_2}, \mathbf{f}_{S_3}$	unitless	1 = valid, 0 = invalid	Star tracker combination
Covariance matrices	$\Sigma_{S_1}, \Sigma_{S_2}, \Sigma_{S_3}, \Sigma_{S_{12}}, \Sigma_{S_{13}}, \Sigma_{S_{23}}, \Sigma_{S_{123}}$	rad <sup>2</sup>	Covariance matrix of combined star tracker attitude	Star tracker combination
Variances	$\sigma_{0,x}^2, \sigma_{0,y}^2, \sigma_{0,z}^2$	rad <sup>2</sup>	Variance function for reconstructed angular rates	Data analysis (cf. Fig. 3)
Half window width	$K$	unitless	Numbers of epochs to the left/right of the centre epoch	
Outputs	Symbol	Unit	Contents	
Attitude quaternions	$\mathbf{q}^{rec}$	unitless	Orientation of GRF wrt. IRF	
Rotation matrices	$\mathbf{R}^{rec}$	unitless	Orientation of GRF wrt. IRF	

Table 15: List of inputs and outputs of attitude reconstruction algorithm

---

**Algorithm 15** Attitude reconstruction algorithm (part 1)
 

---

```

1: Run Algorithm 4 on  $\mathbf{q}^{meas}$ 
2: Interpolate quaternions  $q_n^{meas}$  where  $f_{S_1,n} = f_{S_2,n} = f_{S_3,n} = 0$  using cubic spline interpolation described in Section 2.2. Flags of interpolated quaternions remain zero.
3: for  $n \leftarrow 1, N$  do ▷ Normalize quaternions (including interpolated ones)
4:    $q_n^{meas} = q_n^{meas} / \sqrt{(q_n^{meas})^T q_n^{meas}}$ 
5: end for
6: for  $n \leftarrow 1, N - 1$  do ▷ Calculate rotation quaternions for later use
7:    $\bar{\omega} \leftarrow \frac{\omega_{n+1} + \omega_n}{2} (t_{n+1} - t_n)$ 
8:    $\hat{q}_{n \rightarrow n+1} \leftarrow \begin{bmatrix} \cos(|\bar{\omega}|/2) \\ \frac{\sin(|\bar{\omega}|/2)}{|\bar{\omega}|} \bar{\omega} \end{bmatrix}$  ▷ The hat designates that quaternions are stored
9: end for
10: for  $n \leftarrow 1, N$  do ▷ Reconstruct attitude (exclude first/last  $K$  epochs)
11:   if  $f_{S_1,n} = 0$  and  $f_{S_2,n} = 0$  and  $f_{S_3,n} = 0$  then
12:      $q_n^{rec} = q_n^{meas}$ 
13:      $f_n^{rec} = 0$ 
14:   else
15:      $\Sigma_n^{meas} \leftarrow \Sigma_{S_1}$  or  $\Sigma_{S_2}$  or  $\dots$   $\Sigma_{S_{123}}$  ▷ Depending on  $f_{S_1,n}$ ,  $f_{S_2,n}$  and  $f_{S_3,n}$ 
16:      $\mathbf{N} \leftarrow (\Sigma_n^{meas})^{-1}$  ▷ Initialise normal equation matrix
17:      $\mathbf{b} \leftarrow [0 \ 0 \ 0]^T$  ▷ Initialise right-hand side of normal equations
18:      $q_{n \rightarrow n+k}^{rot} \leftarrow [1 \ 0 \ 0 \ 0]^T$  ▷ Initialise rotation quaternion ( $k = 0$ )
19:     for  $k \leftarrow 1, \min(K, N - n)$  do
20:        $q_{n \rightarrow n+k}^{rot} \leftarrow q_{n \rightarrow n+k}^{rot} \hat{q}_{n+k-1 \rightarrow n+k}^{rot}$ 
21:       if  $f_{S_1,n+k} = 1$  or  $f_{S_2,n+k} = 1$  or  $f_{S_3,n+k} = 1$  then
22:          $q_{n,n+k}^{rel} \leftarrow q_n^{meas} q_{n \rightarrow n+k}^{rot} (q_{n+k}^{meas})^*$ 
23:          $\mathbf{d}_{n,n+k} \leftarrow 2 \begin{bmatrix} q_{n,n+k}^{rel}(2) & q_{n,n+k}^{rel}(3) & q_{n,n+k}^{rel}(4) \end{bmatrix}^T$ 
24:          $\Sigma_{n+k}^{meas} \leftarrow \Sigma_{S_1}$  or  $\dots$   $\Sigma_{S_{123}}$  ▷ Depending on  $f_{S_1,n+k}$ ,  $f_{S_2,n+k}$  and  $f_{S_3,n+k}$ 
25:          $\Sigma_{n+k}^{rot} \leftarrow \begin{bmatrix} \sigma_{x,0}^2 & 0 & 0 \\ 0 & \sigma_{y,0}^2 & 0 \\ 0 & 0 & \sigma_{z,0}^2 \end{bmatrix} (t_n - t_{n+k})^2$ 
26:          $\Sigma_{n+k} \leftarrow \Sigma_{n+k}^{meas} + \Sigma_{n+k}^{rot}$ 
27:          $\mathbf{N} \leftarrow \mathbf{N} + \Sigma_{n+k}^{-1}$ 
28:          $\mathbf{b} \leftarrow \mathbf{b} + \Sigma_{n+k}^{-1} \mathbf{d}_{n,n+k}$ 
29:       end if
30:     end for

```

---




---

**Algorithm 15** Attitude reconstruction algorithm (part 2)

---

```

31:  $q_{n \rightarrow n+k}^{rot} \leftarrow [1 \ 0 \ 0 \ 0]^T$   $\triangleright$  Initialise rotation quaternion ( $k = 0$ )
32: for  $k \leftarrow -1, \max(-K, 1 - n)$  do
33:    $q_{n \rightarrow n+k}^{rot} \leftarrow q_{n \rightarrow n+k}^{rot} (\hat{q}_{n+k \rightarrow n+k+1}^{rot})^*$   $\triangleright$  This line is different to the previous for-loop
34:   if  $f_{S_1, n+k} = 1$  or  $f_{S_2, n+k} = 1$  or  $f_{S_3, n+k} = 1$  then
35:      $q_{n, n+k}^{rel} \leftarrow q_n^{meas} q_{n \rightarrow n+k}^{rot} (q_{n+k}^{meas})^*$ 
36:      $\mathbf{d}_{n, n+k} \leftarrow 2 \left[ q_{n, n+k}^{rel}(2) \ q_{n, n+k}^{rel}(3) \ q_{n, n+k}^{rel}(4) \right]^T$ 
37:      $\Sigma_{n+k}^{meas} \leftarrow \Sigma_{S_1}$  or ...  $\Sigma_{S_{123}}$   $\triangleright$  Depending on  $f_{S_1, n+k}$ ,  $f_{S_2, n+k}$  and  $f_{S_3, n+k}$ 
38:      $\Sigma_{n+k}^{rot} \leftarrow \begin{bmatrix} \sigma_{0,x}^2 & 0 & 0 \\ 0 & \sigma_{0,y}^2 & 0 \\ 0 & 0 & \sigma_{0,z}^2 \end{bmatrix} (t_n - t_{n+k})^2$ 
39:      $\Sigma_{n+k} \leftarrow \Sigma_{n+k}^{meas} + \Sigma_{n+k}^{rot}$ 
40:      $\mathbf{N} \leftarrow \mathbf{N} + \Sigma_{n+k}^{-1}$ 
41:      $\mathbf{b} \leftarrow \mathbf{b} + \Sigma_{n+k}^{-1} \mathbf{d}_{n, n+k}$ 
42:   end if
43: end for
44:  $\tilde{\mathbf{e}} = \mathbf{N}^{-1} \mathbf{b}$ 
45:  $q_n^{rec} \leftarrow \frac{1}{\sqrt{1 + \tilde{\mathbf{e}}^T \tilde{\mathbf{e}}/4}} \begin{bmatrix} 1 \\ -\tilde{\mathbf{e}}/2 \end{bmatrix} q_n^{meas}$   $\triangleright$  Including normalisation of quaternion
46:  $f_n^{rec} = 1$ 
47:  $\mathbf{R}_n^{rec} = \begin{bmatrix} (q_{n,0}^{rec})^2 + (q_{n,1}^{rec})^2 - (q_{n,2}^{rec})^2 - (q_{n,3}^{rec})^2 & 2(q_{n,1}^{rec} q_{n,2}^{rec} + q_{n,0}^{rec} q_{n,3}^{rec}) & 2(q_{n,1}^{rec} q_{n,3}^{rec} - q_{n,0}^{rec} q_{n,2}^{rec}) \\ 2(q_{n,1}^{rec} q_{n,2}^{rec} - q_{n,0}^{rec} q_{n,3}^{rec}) & (q_{n,0}^{rec})^2 - (q_{n,1}^{rec})^2 + (q_{n,2}^{rec})^2 - (q_{n,3}^{rec})^2 & 2(q_{n,2}^{rec} q_{n,3}^{rec} + q_{n,0}^{rec} q_{n,1}^{rec}) \\ 2(q_{n,1}^{rec} q_{n,3}^{rec} + q_{n,0}^{rec} q_{n,2}^{rec}) & 2(q_{n,2}^{rec} q_{n,3}^{rec} - q_{n,0}^{rec} q_{n,1}^{rec}) & (q_{n,0}^{rec})^2 - (q_{n,1}^{rec})^2 - (q_{n,2}^{rec})^2 + (q_{n,3}^{rec})^2 \end{bmatrix}$ 
48:   end if
49: end for

```

---

## 9 Gravity gradient calculation

The calculation of the gravity gradients is provided in Algorithm 16.

Inputs	Symbol	Unit	Contents	Source algorithm
Acceleration	$\mathbf{a}_{d14}, \mathbf{a}_{d25}, \mathbf{a}_{d36}$	$m/s^2$	Calibrated differential mode acceleration	Gradiometer calibration in science mode
Angular rates	$\boldsymbol{\omega}$	$rad/s$	Rotation rate of GRF wrt. IRF	Angular rate reconstruction
Arm length	$L_x, L_y, L_z$	$m$	Lengths of the gradiometer arms	
Outputs	Symbol	Unit	Contents	
Gravity gradients	$V_{xx}, V_{yy}, V_{zz}, V_{xy}, V_{xz}, V_{yz}$	$s^{-2}$	Gravity gradients in GRF	

Table 16: List of inputs and outputs of attitude reconstruction algorithm

---

### Algorithm 16 Gravity gradient calculation algorithm

---

```

1: for  $n \leftarrow 1, N$  do
2:    $V_{xx,n} = -2a_{d14x,n}/L_x - \omega_{y,n}^2 - \omega_{z,n}^2$ 
3:    $V_{yy,n} = -2a_{d25y,n}/L_y - \omega_{x,n}^2 - \omega_{z,n}^2$ 
4:    $V_{zz,n} = -2a_{d36z,n}/L_z - \omega_{x,n}^2 - \omega_{y,n}^2$ 
5:    $V_{xz,n} = -a_{d14z,n}/L_x - a_{d36x,n}/L_z + \omega_{x,n}\omega_{z,n}$ 
6:    $V_{xy,n} = -a_{d25x,n}/L_y - a_{d14y,n}/L_x + \omega_{x,n}\omega_{y,n}$ 
7:    $V_{yz,n} = -a_{d36y,n}/L_z - a_{d25z,n}/L_y + \omega_{y,n}\omega_{z,n}$ 
8: end for

```

---





## References

- P. Groves. *Principles of GNSS, inertial, and multisensor integrated navigation systems*. Artech House, Boston/London, second edition, 2013.
- L. Romans. Optimal combination of quaternions from multiple star cameras, 2003. URL [ftp://podaac.jpl.nasa.gov/allData/grace/docs/quaternion\\_memo.pdf](ftp://podaac.jpl.nasa.gov/allData/grace/docs/quaternion_memo.pdf). Last accessed 31 May 2018.
- C. Stummer, T. Fecher, and R. Pail. Alternative method for angular rate determination within the GOCE gradiometer processing. *J Geod*, 85:585–596, 2011. doi: 10.1007/s00190-011-0461-3.