**Improved magnetometer calibration (Part 2)**

**https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html**

In Part 1 of this post, I have presented the MagCal software that is used to calibrate a 3-axis magnetometer (or accelerometer).

One problem with MagCal is that one of the results (the matrix A) cannot be used directly. You have to calculate the inverse matrix A-1.  The other problem is that the matrix A elements have a limited number of significant digits, which affects the precision of the inverted result. And finally, the raw measurements numbers cannot be greater than 99.99, so that you may have to scale down all your data to use it.

I present here my alternate implementation called Magneto, which calculates directly the required matrix A-1.  It also presents the inverted A matrix to permit a comparison with MagCal results.

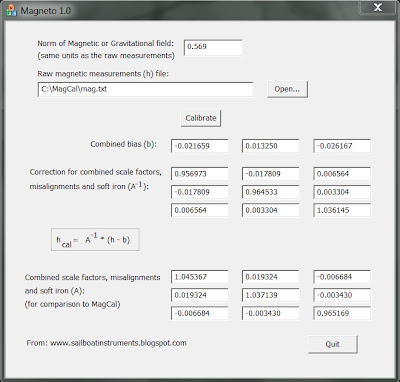
Magneto v1.2 can be downloaded here:

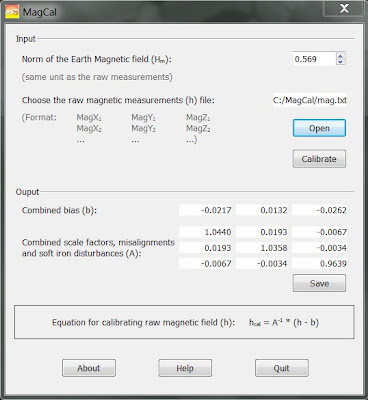
<https://sites.google.com/view/sailboatinstruments1>

What are the differences between MagCal and Magneto?

Both find the equation of an ellipsoid that best fits the raw data points, using however different techniques. MagCal uses an “adaptive least square estimator”, and Magneto uses the “Li's ellipsoid specific fitting algorithm”. Which one is better? I don’t know, but the authors that have developed MagCal suggest in a recent paper that they may now use Li’s method.

Here is the output of both softwares, using the same raw data file provided with MagCal.

[](https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEjs8OcfnRBrWk8F8rgoEFOhUfZjOQsEFHwM5HuNVSVwMLQzgD1qhXgeg0VplqrsWBWef66ykZ9jZWwQIJqrmr9r2M0jkqgyWBBXrhrZPPKXkZilPQCvqE6ynmK01lNVeQUWegRvwG-EoYbu/s1600/magneto.jpg)

[](https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEgZ-bjsc-tImRg53q76i1vwM8PCgHQW8PkmaDE-DVQfgN_dsNn_DMHaN8tCLm15f1vKTORo4AmHW_DTPpK4Kjw_ZjNTHxngnTgSf9YdyBHLmjXmzUmEZGICjIuo75xFEDkcjvIbUNJZwBQS/s1600/magcalcomp.jpg)

As you can see, the results are nearly identical. I plan to add new features and to fully document Magneto on its hosting site when time permits.

Posted by Merlin at [10:48 PM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html)

[Email This](https://www.blogger.com/share-post.g?blogID=4237678261594285909&postID=8799581456670244109&target=email)[BlogThis!](https://www.blogger.com/share-post.g?blogID=4237678261594285909&postID=8799581456670244109&target=blog)[Share to Twitter](https://www.blogger.com/share-post.g?blogID=4237678261594285909&postID=8799581456670244109&target=twitter)[Share to Facebook](https://www.blogger.com/share-post.g?blogID=4237678261594285909&postID=8799581456670244109&target=facebook)[Share to Pinterest](https://www.blogger.com/share-post.g?blogID=4237678261594285909&postID=8799581456670244109&target=pinterest)

**23 comments:**

1. https://resources.blogblog.com/img/blank.gif

[**Calibration**](http://bestandards.com/)[November 21, 2011 at 12:42 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1321854137759#c6199315999934016348)

Hi there, awesome site. I thought the topics you posted on were very interesting. I tried to add your RSS to my feed reader and it a few. take a look at it, hopefully I can add you and follow.

[Reply](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**Anonymous**[January 10, 2012 at 4:51 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1326189118558#c598758438080604701)

Hello,  
Is it possible to modify the source code ? Because I would like to implement the algorithm in an embedded system.  
I can't save a lot of data in a file like in a pc, I have to execute the algorithm with only few data.

[Reply](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

[**Merlin**](https://www.blogger.com/profile/00901116173524809046)[January 10, 2012 at 9:24 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1326205447096#c5514131380974161093)

It is certainly possible, if you can find or develop an alternate math library (instead of Fortran) for matrix calculations. Matrix multiplication and inversion code are readily available, eigenvalues calculations may be harder to come by. I would strongly consider a 32-bit microprocessor for such a task.

[Reply](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

[**Merlin**](https://www.blogger.com/profile/00901116173524809046)[January 23, 2012 at 9:19 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1327328363333#c1722076224698383521)

For those who have inquired on the possibility of porting the Magneto source code to an embedded system, here is a site with extensive matrix source code in C language:  
  
http://www.mymathlib.com/matrices/  
  
Be aware that there is an important difference in the way the matrix elements are stored in memory between Fortran and C. In Fortran, the elements are stored column by column, while in C and C++, they are stored line by line.  
  
And the Moore-Penrose pseudo-inverse may also be replaced by the normal matrix inverse, as they are normally the same for a square matrix, unless your data set is bad or incomplete.

[Reply](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

[**Merlin**](https://www.blogger.com/profile/00901116173524809046)[January 25, 2012 at 10:05 PM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1327547140882#c4461823690837132455)

I have taken a step further in porting the Magneto source code to 100% C-language. It is available at:  
  
https://sites.google.com/site/sailboatinstruments1/c-language-implementation

[Reply](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**Anonymous**[April 22, 2012 at 1:01 PM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1335114105821#c3959689081313592051)

Very nice work. Helps me a lot to write a calibration routine for my lowcost MEMS-Magnetometer. Thank you for sharing.

[Reply](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**Anonymous**[April 24, 2012 at 3:12 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1335251554471#c4441296973354471746)

Hi Mr. Merlin  
  
I am a bit confused about the input data. When you say 'raw data', does it mean the data come from ADC (whose unit is 'counts') ?  
I've looked into your sample data (sample.txt), and i found the value of all data are less than 1 and greater than -1 .  
How did you get this 'raw data' ?  
do we have to calibrate the ADC output first using parameter provided in the datasheet?  
  
Thank you for your support.

[Reply](javascript:;)

[Replies](javascript:;)

* 1. https://www.blogger.com/img/blogger_logo_round_35.png

[**Merlin**](https://www.blogger.com/profile/00901116173524809046)[April 24, 2012 at 10:12 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1335276764536#c9074243846329085850)

Magneto expects to receive raw data in +- format (a value of zero indicating a null field in the current axis), but not necessarily normalized to +-1.0.  
  
If your sensors have SPI or I2C outputs, they will usually directly produce the required format. For example, the MicroMag3 magnetometer directly produces counts from -3411 to +3411, and the the SCA3000 accelerometer directly produces counts from -1333 to 1333, and Magneto can process direcly these values, without the need to normalize them to +- 1.0. I understand that a normalization may be desirable to avoid machine precision problems, but this has not been the case with these sensors.  
  
If your sensors produce voltage levels that you have to convert to counts with an ADC, you have indeed to substract a zero field value from the ADC output before using Magneto. You would then normally choose the maximum positive value as input to the 'Norm of Magnetic or Gravitational field'.  
  
But this norm value is not critical if all you want to calculate later on is a heading (if it is a magnetometer) or a tilt angle (if it is an accelerometer). You can input any reasonable value for the norm, the correction matrix will be different by just a scaling factor, but the calculated heading (or tilt angle) will be the same, as it depends only on the relative value of the field components. The bias values will be unchanged, as they do not depend on the norm.

[**Reply**](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**Anonymous**[November 6, 2014 at 12:11 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1415250689738#c1530438358536001979)

Could I ask you to explain how you find the value "norm of magnetic or gravitational field" please? This has a big effect on the results and I don't know how to find this value for the program. Thank you.

[Reply](javascript:;)

[Replies](javascript:;)

* 1. https://www.blogger.com/img/blogger_logo_round_35.png

[**Merlin**](https://www.blogger.com/profile/00901116173524809046)[November 6, 2014 at 9:50 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1415285421667#c3452072027113043092)

I am quoting a part of a previous answer as it is directly relevant to your question:  
"You would then normally choose the maximum positive value as input to the 'Norm of Magnetic or Gravitational field'.  
  
But this norm value is not critical if all you want to calculate later on is a heading (if it is a magnetometer) or a tilt angle (if it is an accelerometer). You can input any reasonable value for the norm, the correction matrix will be different by just a scaling factor, but the calculated heading (or tilt angle) will be the same, as it depends only on the relative value of the field components. The bias values will be unchanged, as they do not depend on the norm."  
In other words, inspect your output values and use the largest, which will likely represents the field magnitude when the sensor is aligned with it.

[**Reply**](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**DrSAR**[November 19, 2014 at 1:17 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1416377875642#c8642081037998693051)

I am messing around with your C-based calibration tool magneto and am trying to feed it some simple data such as  
1 0 0  
0 1 0  
0 0 1  
-1 0 0  
0 -1 0  
0 0 -1  
.707106781 .707106781 0  
0 .707106781 .707106781  
.707106781 0 .707106781  
  
I thought these are sampled on a perfect sphere. As expected I get a bias of 0 in all three axis. I would have expected to obtain an A^-1 as the identity. This is not so. What am I doing wrong?

[Reply](javascript:;)

[Replies](javascript:;)

* 1. https://resources.blogblog.com/img/blank.gif

**DrSAR**[November 19, 2014 at 1:49 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1416379763199#c1617362018675196278)

Ignore me. You actually do get (almost) the identity. Since the data is on a sphere, the orientation for the elipsoid is not well defined. So it appears to be rotated by some (arbitrary?) orientation but still gives you the expected values h\_cal for a choice if input h. A further mistake I made was to ignore the variable hm in the code which encodes for the above mentioned 'norm of magnetic or gravitational field'.

[**Reply**](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

[**messari omar**](https://www.blogger.com/profile/02094894726024733610)[April 25, 2018 at 9:27 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1524662869015#c1805690910088822860)

i really want to do the same stuff but on the C# can you give me the algorithm of your code it have been 2 weeks i'm working on it and didn't found anything yet.  
  
Can you help please

[Reply](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

[**Mfweb**](https://www.blogger.com/profile/01426355019314661242)[August 13, 2019 at 2:32 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1565677935114#c173126483582213712)

I got A-1 and b in the formula, how can I use it in a real application? Is there a related tutorial?

[Reply](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

[**Paul**](https://www.blogger.com/profile/05976197158116330147)[March 29, 2022 at 1:59 PM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1648576779977#c3252722383524850078)

This comment has been removed by the author.

[Reply](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

[**风吹屁档凉飕飕**](https://www.blogger.com/profile/03656461936953135425)[June 26, 2022 at 1:38 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1656221886731#c4854010743814239371)

Thx for your sharing.

[Reply](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**Anonymous**[August 17, 2022 at 1:18 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1660713522154#c3078937678852755803)

Nice Post...........  
  
[Buy BILL COUNTER machine in Cambodia](https://moneycounterchina.com/product-category/money-counter-banknote-counter-currency-counter-ticket-counter/), [Buy Euro USD GBP SAR LBP MYR KHR value counter machine](https://moneycounterchina.com/product-category/cis-value-counting-machine-mix-denomination-money-counter/euro-mix-value-counting-machine/), [Buy ECB Teste Euro Fake Note Money Detector in Europe](https://moneycounterchina.com/product-category/infrared-money-detector/), [Buy Two Pocket Banknote Sorting Machine for USD EUR GBP Multi Currency Money Sorter](https://moneycounterchina.com/product-category/cis-value-counting-machine-mix-denomination-money-counter/iqd-mix-value-counter/), [Mix Value Counting Machine in Laos](https://moneycounterchina.com/product-category/cis-value-counting-machine-mix-denomination-money-counter/cad-mix-value-counter/)

[Reply](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**Anonymous**[August 19, 2023 at 4:58 PM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1692478680239#c2208603149241447180)

I'm interested in your 100% C-language

[Reply](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**Anonymous**[August 19, 2023 at 4:59 PM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1692478748291#c8455796809426906063)

I'm interested in your 100% C-language port of the Magneto code, but keep getting errors at the Google URL. Is there any other way to see the file(s)?

[Reply](javascript:;)

[Replies](javascript:;)

* 1. https://resources.blogblog.com/img/blank.gif

**E. Clark**[September 22, 2023 at 9:06 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1695388011706#c1335455046229064824)

I am also interested and getting the error back from the link.

[**Reply**](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

[**Designessential**](https://www.blogger.com/profile/10709201350627107703)[July 29, 2024 at 4:06 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1722240373043#c1484916392901408682)

The expert[home interior designers](https://designessentials.co.in/) The team is dedicated to transforming your spaces into stunning, functional, and personalized havens.

[Reply](javascript:;)

1. https://resources.blogblog.com/img/blank.gif

**Anonymous**[September 2, 2024 at 3:40 PM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1725306029585#c711291002805256129)

Respect and I have a tremendous proposal: Who To Contact For House Renovation [house renovation price](https://groups.google.com/g/homerenotoronto/c/9cCt-ELbJ6g)

[Reply](javascript:;)

1. https://www.blogger.com/img/blogger_logo_round_35.png

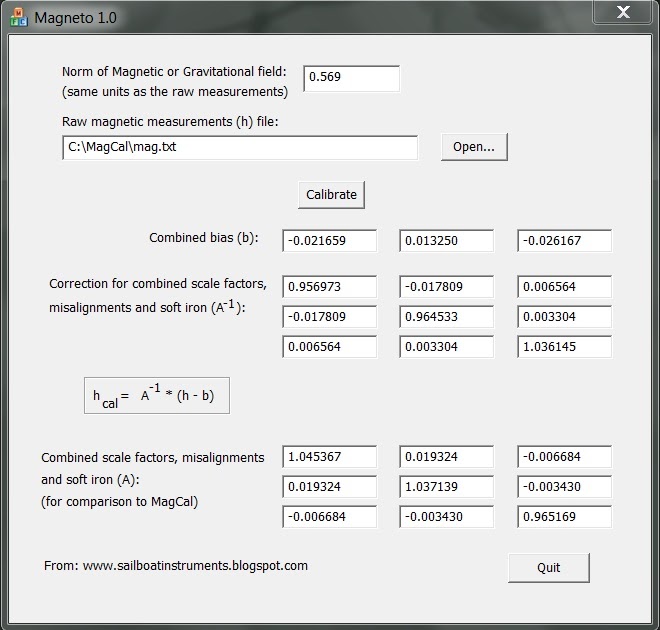
[**Designessential**](https://www.blogger.com/profile/10709201350627107703)[September 23, 2024 at 1:11 AM](https://sailboatinstruments.blogspot.com/2011/09/improved-magnetometer-calibration-part.html?showComment=1727068283143#c4365302054645001171)

Design Essentials in Delhi, NCR- [Front Elevation Design](https://designessentials.co.in/front-elevation-design/) We are the Best Interior Designer In Commercial And residential areas in Delhi NCR.

[Reply](javascript:;)

[sailboatinstruments1](https://sites.google.com/view/sailboatinstruments1/a-download-magneto-v1-2)

# A. Download Magneto v1.2



If you would like to help in current and future projects on this site and its companion blog ([www.sailboatinstruments.blogspot.com](http://www.google.com/url?q=http%3A%2F%2Fwww.sailboatinstruments.blogspot.com&sa=D&sntz=1&usg=AOvVaw3lKxPFrdseVDqFAsB8RpWZ)), please consider a donation. Money will be used to acquire hardware for developing more custom sailboat instruments and software.

[[https://lh6.googleusercontent.com/dyXRui6bwBGX7MZCZKw0vYZWtMa5cNbMJxeeS2iPwZaGtwr76cwQjFm8fqGhEv2QxvwpEEhht_Qwm_FWuRycQ5VTarcGxbh8EgOZhSkcjsx9JiRmfozHKOYJaZ4-nt1Fyw=w1280](https://www.google.com/url?q=https%3A%2F%2Fwww.paypal.com%2Fcgi-bin%2Fwebscr%3Fcmd%3D_donations%26business%3Dmboulanger%2540oricom%252eca%26lc%3DCA%26item_name%3DSailboat%2520Instruments%26currency_code%3DUSD%26bn%3DPP%252dDonationsBF%253abtn_donateCC_LG%252egif%253aNonHosted&sa=D&sntz=1&usg=AOvVaw2wjkgm2ttG78v-_RJYo7iU)](https://www.google.com/url?q=https%3A%2F%2Fwww.paypal.com%2Fcgi-bin%2Fwebscr%3Fcmd%3D_donations%26business%3Dmboulanger%2540oricom%252eca%26lc%3DCA%26item_name%3DSailboat%2520Instruments%26currency_code%3DUSD%26bn%3DPP%252dDonationsBF%253abtn_donateCC_LG%252egif%253aNonHosted&sa=D&sntz=1&usg=AOvVaw2wjkgm2ttG78v-_RJYo7iU" \t "_blank)

[sailboatinstruments1](https://sites.google.com/view/sailboatinstruments1/a-download-magneto-v1-2)

# B. Presentation

While searching the subject of magnetometer calibration, I came upon [MagCal](http://www.google.com/url?q=http%3A%2F%2Fsailboatinstruments.blogspot.com%2F2011%2F08%2Fimproved-magnetometer-calibration.html&sa=D&sntz=1&usg=AOvVaw2je5QWxTWprWLkFx9T__U3), a software tool developed by the Position, Location And Navigation Group (PLAN) at the University of Calgary. I got the idea of developing my own implementation of such a tool, as part of the development of a custom compass.

Magneto, like MagCal, looks for the ellipsoid that best fits the magnetometer raw measurements. Magneto uses a technique known as "Li's ellipsoid specific fitting algorithm", described in [this paper](http://www.google.com/url?q=http%3A%2F%2Fwww.computer.org%2Fportal%2Fweb%2Fcsdl%2Fabs%2Fproceedings%2Fgmp%2F2004%2F2078%2F00%2F20780335abs.htm&sa=D&sntz=1&usg=AOvVaw2seQMX6517XSBSzzXsa-Hw), and for which a [Matlab implementation](http://www.google.com/url?q=http%3A%2F%2Fwww.mathworks.com%2Fmatlabcentral%2Ffileexchange%2F23377-ellipsoid-fitting&sa=D&sntz=1&usg=AOvVaw0W7NdhnAxViZmTyufWTsfq) is available.

After the characteristics of the ellipsoid are established, a correction matrix can be developed. This [Application Note](http://www.google.com/url?q=http%3A%2F%2Fwww.freescale.com%2Ffiles%2Fsensors%2Fdoc%2Fapp_note%2FAN4246.pdf&sa=D&sntz=1&usg=AOvVaw0CnpxbdxwIrgOtsALoBfEl) has been a useful reference for this step.

[sailboatinstruments1](https://sites.google.com/view/sailboatinstruments1/a-download-magneto-v1-2)

# C. The Mathematical Toolkit

Magneto is a Microsoft Foundation Class (MFC) dialog-based application using [LAPACK for Windows](http://www.google.com/url?q=http%3A%2F%2Ficl.cs.utk.edu%2Flapack-for-windows%2Flapack%2F&sa=D&sntz=1&usg=AOvVaw1djR-mDoYdSmmENJYXtJgp) Fortran libraries. The following LAPACK functions are used for matrix manipulations:

* + [DGEMM](http://www.google.com/url?q=http%3A%2F%2Fwww.netlib.org%2Fblas%2Fdgemm.f&sa=D&sntz=1&usg=AOvVaw1W2X9OrK6K8uuiqh6IlxlP) (for matrix multiplication)
  + [DGELSS](http://www.google.com/url?q=http%3A%2F%2Fwww.netlib.org%2Flapack%2Fdouble%2Fdgelss.f&sa=D&sntz=1&usg=AOvVaw3DLHxeRvLgOWHclvIfNfTv) (to calculate a matrix pseudo-inverse)
  + [DGETRF](http://www.google.com/url?q=http%3A%2F%2Fwww.netlib.org%2Flapack%2Fdouble%2Fdgetrf.f&sa=D&sntz=1&usg=AOvVaw3Uy6QYWmPkNXy7GI87nS6X) and [DGETRI](http://www.google.com/url?q=http%3A%2F%2Fwww.netlib.org%2Flapack%2Fdouble%2Fdgetri.f&sa=D&sntz=1&usg=AOvVaw0eVU5dRhR4t5A1Rr6pY8vK) (to calculate a matrix inverse)
  + [DGEEV](http://www.google.com/url?q=http%3A%2F%2Fwww.netlib.org%2Flapack%2Fdouble%2Fdgeev.f&sa=D&sntz=1&usg=AOvVaw2opVYM26Fe4QHCSjE2TnJU) (to calculate eigenvalues and eigenvectors)

extern "C"

{

void DGEMM(const char \*transa, const char \*transb, const int \*m, const int \*n, const int \*k,

const double \*alpha, const double \*a, const int \*lda, const double \*b, const int \*ldb,

const double\* beta, double \*c, const int \*ldc);

void DGELSS(int \*m, int \*n, int \*nrhs, double \*a, int \*lda, double \*b, int \*ldb, double \*sOUT,

double \*rcondIN, int \*rankOUT, double \*work, int \*lwork, int \*infoOUT);

void DGETRF(const int \*m, const int \*n, double \*a, const int \*lda, double \*ipiv, int \*info);

void DGETRI(const int \*n, double \*a, const int \*lda, double \*ipiv, double \*work, const int \*lwork, int \*info);

void DGEEV(const char \*jobvl, const char \*jobvr, const int \*n, double \*a, const int \*lda, double \*wr,

double \*wi, double \*vl, const int \*ldvl, double \*vr, const int \*ldvr, double \*work,

const int \*lwork, int \*info);

}

In debug mode, link to:

BLASd.lib

LAPACKd.lib

In release mode, link to:

BLAS.lib

LAPACK.lib

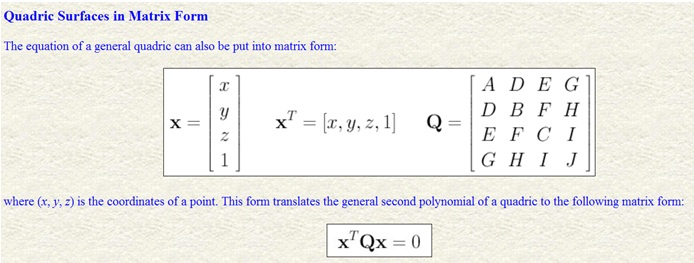
[sailboatinstruments1](https://sites.google.com/view/sailboatinstruments1/a-download-magneto-v1-2)

# E. Geometric interpretation

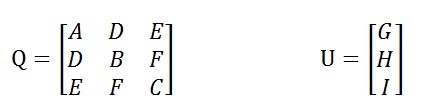
An ellipsoid can be described by its general equation:

Ax2 + By2 + Cz2 + 2Dxy + 2Exz + 2Fyz + 2Gx + 2Hy + 2Iz + J = 0

In matrix form:



If we define



then the center of the ellipsoid can be calculated as the vector **b**= - **Q**-1 **u**

The ellipsoid's semi-axis a, b and c are given by:

https://lh5.googleusercontent.com/Twc5k-tztUO-ph28n6QFbbdLm3B36bjOZ6a_M94x6Sn-uG8TeuA6brAu5H5U29SK9YyFBKbbSAUoP0FEwduxoCY3xlumevljubUQ-kFr9ijNDw1LtniJj7JjENfk_QxZDA=w1280

where : λ1, λ2 and λ3 are the eigenvalues of **Q**.

The corresponding normalized eigenvectors **v1**, **v2** and **v3** of the matrix **Q** describe the direction of the 3 principal axis of the ellipsoid. And the matrix **V** formed from the column vectors **v1**, **v2** and **v3** is the rotation matrix that describes the orientation of the ellipsoid in the reference frame.

**Mapping a general ellipsoid to a sphere of radius *R* centered at (0,0)**

This will be done in 5 steps:

Step 1. Translate the ellipsoid so that its center coincides with the origin(0,0)

Step 2. Align the principal axis of the ellipsoid with the x, y and z-axis of the reference frame

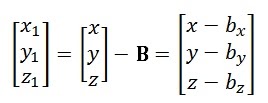
Step 3. Scale the principal axis of the ellipsoid so that they all have the same length (becoming a sphere)

Step 4. Rotate back the sphere to the original orientation of the ellipsoid

Step 5. Scale the sphere to a norm R

For any point **x** (x,y,z) at the surface of the ellipsoid:

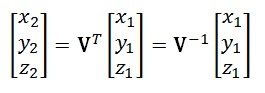
**Step 1 :** The center of the ellipsoid is transfered to the origin (0,0) of the reference frame. The point **x** becomes **x1**.

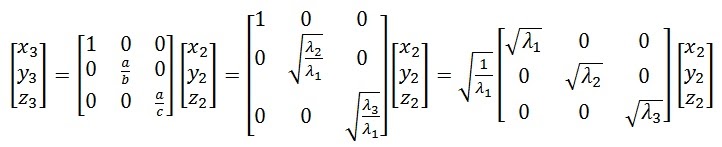


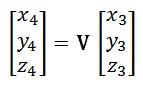
**Step 4 :** the sphere is rotated back to the original orientation of the ellipsoid by the rotation matrix **V**. The point **x3** becomes **x4**.

**Step 3 :** The semi-axis b and c are scaled to the length of the semi-axis a, transforming the ellipsoid into a sphere of radius a. The point **x2** becomes **x3**.

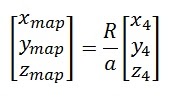
**Step 2 :** The principal axis of the ellipsoid are aligned with the x, y and z-axis of the reference frame. This corresponds to applying an inverse rotation to that described by **V**, that is **V**T or **V**-1 as the inverse and transpose of a rotation matrix are the same. The point **x1** becomes **x2**.



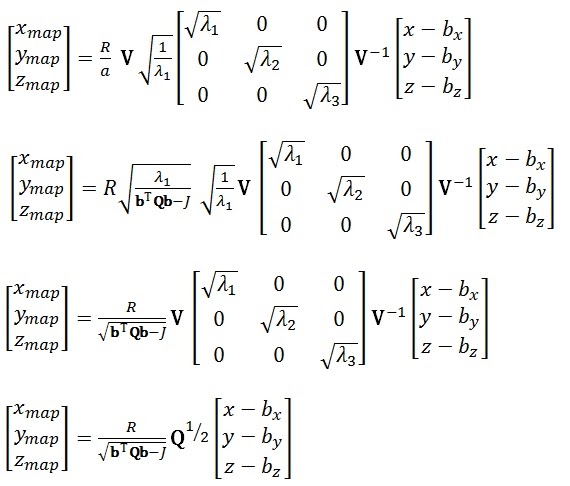




**Step 5 :** the radius of the sphere is changed from a to R. The point **x4** becomes **xmap**.



**Summing it all (Step 1 to 5) :**



[sailboatinstruments1](https://sites.google.com/view/sailboatinstruments1/a-download-magneto-v1-2)

# G. C-language implementation

This implementation makes use of the C-language matrix source code available at

[http://www.mymathlib.com/matrices/](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2F&sa=D&sntz=1&usg=AOvVaw2heDPrcfuhCpq7ElW-_UxX)

It has been tested as a console application, and reproduces exactly the results given by the FORTRAN implementation of Magneto 1.2, using the test file mag.txt with a user norm of 0.569.

Copyright (C) 2013 [www.sailboatinstruments.blogspot.com](http://www.google.com/url?q=http%3A%2F%2Fwww.sailboatinstruments.blogspot.com&sa=D&sntz=1&usg=AOvVaw3lKxPFrdseVDqFAsB8RpWZ)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

#include <stdio.h>

#include <math.h>

#include <malloc.h>

#include <string.h>

#include <float.h>

// Forward declarations of mymathlib.com routines

void[Matrix\_x\_Its\_Transpose](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Farithmetic%2FAtB_ABt_AtA_AAt.html&sa=D&sntz=1&usg=AOvVaw3wmpjtydZ-SEHjKCLCOrL-)(double\*, double\*, int, int);

void[Get\_Submatrix](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Fdatamovement%2Fget_set_submatrix.html&sa=D&sntz=1&usg=AOvVaw0x_y6ms1xRGmiNqYhOpLu4)(double\*, int, int, double\*, int, int, int);

int[Choleski\_LU\_Decomposition](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Flinearsystems%2Fcholeski.html&sa=D&sntz=1&usg=AOvVaw3nnsm7AW4Bc8sx7F-IcyA8)(double\*, int);

int[Choleski\_LU\_Inverse](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Flinearsystems%2Fcholeski.html&sa=D&sntz=1&usg=AOvVaw3nnsm7AW4Bc8sx7F-IcyA8)(double\*, int);

void[Multiply\_Matrices](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Farithmetic%2Fmul_mat.html&sa=D&sntz=1&usg=AOvVaw0Xh9qTbHFSvMr0dJOBGR0M)(double\*, double\*, int, int, double\*, int);

void[Identity\_Matrix](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Farithmetic%2Fconstants.html&sa=D&sntz=1&usg=AOvVaw3sJ0PMjdpxUNHk-qaadDWD)(double\*, int);

int[Hessenberg\_Form\_Elementary](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Feigen%2Fhessenberg.html&sa=D&sntz=1&usg=AOvVaw0WQ7cMvhnzjmzVRSomAX7e)(double\*, double\*, int);

static void Hessenberg\_Elementary\_Transform(double\*, double\*, int[], int);

void[Copy\_Vector](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Fdatamovement%2Fcopy_matrix.html&sa=D&sntz=1&usg=AOvVaw3BRA9WyWiApHd2KFul74F6)(double\*, double\*, int);

int[QR\_Hessenberg\_Matrix](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Feigen%2Fgeneral.html&sa=D&sntz=1&usg=AOvVaw3wa3HFbRAo_E3RYckZsNWf)(double\*, double\*, double[], double[], int, int);

static void One\_Real\_Eigenvalue(double[], double[], double[], int, double);

static void Two\_Eigenvalues(double\*, double\*, double[], double[], int, int, double);

static void Update\_Row(double\*, double, double, int, int);

static void Update\_Column(double\*, double, double, int, int);

static void Update\_Transformation(double\*, double, double, int, int);

static void Double\_QR\_Iteration(double\*, double\*, int, int, int, double\*, int);

static void Product\_and\_Sum\_of\_Shifts(double\*, int, int, double\*, double\*, double\*, int);

static int Two\_Consecutive\_Small\_Subdiagonal(double\*, int, int, int, double, double);

static void Double\_QR\_Step(double\*, int, int, int, double, double, double\*, int);

static void BackSubstitution(double\*, double[], double[], int);

static void BackSubstitute\_Real\_Vector(double\*, double[], double[], int, double, int);

static void BackSubstitute\_Complex\_Vector(double\*, double[], double[], int, double, int);

static void Calculate\_Eigenvectors(double\*, double\*, double[], double[], int);

static void Complex\_Division(double, double, double, double, double\*, double\*);

void[Transpose\_Square\_Matrix](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2Fdatamovement%2Ftransposing.html&sa=D&sntz=1&usg=AOvVaw2Bx0Hr3Q70tA2Xk02DIieS)(double\*, int);

int main()

{

int nlines = 0;

char buf[120];

double \*D, \*S, \*C, \*S11, \*S12, \*S12t, \*S22, \*S22\_1, \*S22a, \*S22b, \*SS, \*E, \*d, \*U, \*SSS;

double \*eigen\_real, \*eigen\_imag, \*v1, \*v2, \*v, \*Q, \*Q\_1, \*B, \*QB, J, hmb, \*SSSS;

int \*p;

int i, index;

double maxval, norm, btqb, \*eigen\_real3, \*eigen\_imag3, \*Dz, \*vdz, \*SQ, \*A\_1, hm, norm1, norm2, norm3;

double x, y, z;

FILE \*fp;

fp = fopen("c:\\MagCal\\mag.txt", "r");

while(fgets(buf, 100, fp) != NULL)

nlines++;

rewind(fp);

D = (double\*)malloc(10 \* nlines \* sizeof(double));

for( i = 0; i < nlines; i++)

{

fgets(buf, 100, fp);

sscanf(buf, "%lf\t%lf\t%lf", &x, &y, &z);

D[i] = x \* x;

D[nlines+i] = y \* y;

D[nlines\*2+i] = z \* z;

D[nlines\*3+i] = 2.0 \* y \* z;

D[nlines\*4+i] = 2.0 \* x \* z;

D[nlines\*5+i] = 2.0 \* x \* y;

D[nlines\*6+i] = 2.0 \* x;

D[nlines\*7+i] = 2.0 \* y;

D[nlines\*8+i] = 2.0 \* z;

D[nlines\*9+i] = 1.0;

}

fclose(fp);

// allocate memory for matrix S

S = (double\*)malloc(10 \* 10 \* sizeof(double));

Matrix\_x\_Its\_Transpose(S, D, 10, nlines);

// Create pre-inverted constraint matrix C

C = (double\*)malloc(6 \* 6 \* sizeof(double));

C[0] = 0.0; C[1] = 0.5; C[2] = 0.5; C[3] = 0.0; C[4] = 0.0; C[5] = 0.0;

C[6] = 0.5; C[7] = 0.0; C[8] = 0.5; C[9] = 0.0; C[10] = 0.0; C[11] = 0.0;

C[12] = 0.5; C[13] = 0.5; C[14] = 0.0; C[15] = 0.0; C[16] = 0.0; C[17] = 0.0;

C[18] = 0.0; C[19] = 0.0; C[20] = 0.0; C[21] = -0.25; C[22] = 0.0; C[23] = 0.0;

C[24] = 0.0; C[25] = 0.0; C[26] = 0.0; C[27] = 0.0; C[28] = -0.25; C[29] = 0.0;

C[30] = 0.0; C[31] = 0.0; C[32] = 0.0; C[33] = 0.0; C[34] = 0.0; C[35] = -0.25;

S11 = (double\*)malloc(6 \* 6 \* sizeof(double));

Get\_Submatrix(S11, 6, 6, S, 10, 0, 0);

S12 = (double\*)malloc(6 \* 4 \* sizeof(double));

Get\_Submatrix(S12, 6, 4, S, 10, 0, 6);

S12t = (double\*)malloc(4 \* 6 \* sizeof(double));

Get\_Submatrix(S12t, 4, 6, S, 10, 6, 0);

S22 = (double\*)malloc(4 \* 4 \* sizeof(double));

Get\_Submatrix(S22, 4, 4, S, 10, 6, 6);

S22\_1 = (double\*)malloc(4 \* 4 \* sizeof(double));

for(i = 0; i < 16; i++)

S22\_1[i] = S22[i];

Choleski\_LU\_Decomposition(S22\_1, 4);

Choleski\_LU\_Inverse(S22\_1, 4);

// Calculate S22a = S22\_1 \* S12t 4\*6 = 4x4 \* 4x6 C = AB

S22a = (double\*)malloc(4 \* 6 \* sizeof(double));

Multiply\_Matrices(S22a, S22\_1, 4, 4, S12t, 6);

// Then calculate S22b = S12 \* S22a ( 6x6 = 6x4 \* 4x6)

S22b = (double\*)malloc(6 \* 6 \* sizeof(double));

Multiply\_Matrices(S22b, S12, 6, 4, S22a, 6);

// Calculate SS = S11 - S22b

SS = (double\*)malloc(6 \* 6 \* sizeof(double));

for(i = 0; i < 36; i++)

SS[i] = S11[i] - S22b[i];

E = (double\*)malloc(6 \* 6 \* sizeof(double));

Multiply\_Matrices(E, C, 6, 6, SS, 6);

SSS = (double\*)malloc(6 \* 6 \* sizeof(double));

Hessenberg\_Form\_Elementary(E, SSS, 6);

eigen\_real = (double\*)malloc(6 \* sizeof(double));

eigen\_imag = (double\*)malloc(6 \* sizeof(double));

QR\_Hessenberg\_Matrix(E, SSS, eigen\_real, eigen\_imag, 6, 100);

index = 0;

maxval = eigen\_real[0];

for(i = 1; i < 6; i++)

{

if(eigen\_real[i] > maxval)

{

maxval = eigen\_real[i];

index = i;

}

}

v1 = (double\*)malloc(6 \* sizeof(double));

v1[0] = SSS[index];

v1[1] = SSS[index+6];

v1[2] = SSS[index+12];

v1[3] = SSS[index+18];

v1[4] = SSS[index+24];

v1[5] = SSS[index+30];

// normalize v1

norm = sqrt(v1[0] \* v1[0] + v1[1] \* v1[1] + v1[2] \* v1[2] + v1[3] \* v1[3] + v1[4] \* v1[4] + v1[5] \* v1[5]);

v1[0] /= norm;

v1[1] /= norm;

v1[2] /= norm;

v1[3] /= norm;

v1[4] /= norm;

v1[5] /= norm;

if(v1[0] < 0.0)

{

v1[0] = -v1[0];

v1[1] = -v1[1];

v1[2] = -v1[2];

v1[3] = -v1[3];

v1[4] = -v1[4];

v1[5] = -v1[5];

}

// Calculate v2 = S22a \* v1 ( 4x1 = 4x6 \* 6x1)

v2 = (double\*)malloc(4 \* sizeof(double));

Multiply\_Matrices(v2, S22a, 4, 6, v1, 1);

v = (double\*)malloc(10 \* sizeof(double));

v[0] = v1[0];

v[1] = v1[1];

v[2] = v1[2];

v[3] = v1[3];

v[4] = v1[4];

v[5] = v1[5];

v[6] = -v2[0];

v[7] = -v2[1];

v[8] = -v2[2];

v[9] = -v2[3];

Q = (double\*)malloc(3 \* 3 \* sizeof(double));

Q[0] = v[0];

Q[1] = v[5];

Q[2] = v[4];

Q[3] = v[5];

Q[4] = v[1];

Q[5] = v[3];

Q[6] = v[4];

Q[7] = v[3];

Q[8] = v[2];

U = (double\*)malloc(3 \* sizeof(double));

U[0] = v[6];

U[1] = v[7];

U[2] = v[8];

Q\_1 = (double\*)malloc(3 \* 3 \* sizeof(double));

for(i = 0; i < 9; i++)

Q\_1[i] = Q[i];

Choleski\_LU\_Decomposition(Q\_1, 3);

Choleski\_LU\_Inverse(Q\_1, 3);

// Calculate B = Q-1 \* U ( 3x1 = 3x3 \* 3x1)

B = (double\*)malloc(3 \* sizeof(double));

Multiply\_Matrices(B, Q\_1, 3, 3, U, 1);

B[0] = -B[0]; // x-axis combined bias

B[1] = -B[1]; // y-axis combined bias

B[2] = -B[2]; // z-axis combined bias

for(i = 0; i < 3; i++)

printf("%lf\r\n", B[i]);

// First calculate QB = Q \* B ( 3x1 = 3x3 \* 3x1)

QB = (double\*)malloc(3 \* sizeof(double));

Multiply\_Matrices(QB, Q, 3, 3, B, 1);

// Then calculate btqb = BT \* QB ( 1x1 = 1x3 \* 3x1)

Multiply\_Matrices(&btqb, B, 1, 3, QB, 1);

// Calculate hmb = sqrt(btqb - J).

J = v[9];

hmb = sqrt(btqb - J);

// Calculate SQ, the square root of matrix Q

SSSS = (double\*)malloc(3 \* 3 \* sizeof(double));

Hessenberg\_Form\_Elementary(Q, SSSS, 3);

eigen\_real3 = (double\*)malloc(3 \* sizeof(double));

eigen\_imag3 = (double\*)malloc(3 \* sizeof(double));

QR\_Hessenberg\_Matrix(Q, SSSS, eigen\_real3, eigen\_imag3, 3, 100);

// normalize eigenvectors

norm1 = sqrt(SSSS[0] \* SSSS[0] + SSSS[3] \* SSSS[3] + SSSS[6] \* SSSS[6]);

SSSS[0] /= norm1;

SSSS[3] /= norm1;

SSSS[6] /= norm1;

norm2 = sqrt(SSSS[1] \* SSSS[1] + SSSS[4] \* SSSS[4] + SSSS[7] \* SSSS[7]);

SSSS[1] /= norm2;

SSSS[4] /= norm2;

SSSS[7] /= norm2;

norm3 = sqrt(SSSS[2] \* SSSS[2] + SSSS[5] \* SSSS[5] + SSSS[8] \* SSSS[8]);

SSSS[2] /= norm3;

SSSS[5] /= norm3;

SSSS[8] /= norm3;

Dz = (double\*)malloc(3 \* 3 \* sizeof(double));

for(i = 0; i < 9; i++)

Dz[i] = 0.0;

Dz[0] = sqrt(eigen\_real3[0]);

Dz[4] = sqrt(eigen\_real3[1]);

Dz[8] = sqrt(eigen\_real3[2]);

vdz = (double\*)malloc(3 \* 3 \* sizeof(double));

Multiply\_Matrices(vdz, SSSS, 3, 3, Dz, 3);

Transpose\_Square\_Matrix(SSSS, 3);

SQ = (double\*)malloc(3 \* 3 \* sizeof(double));

Multiply\_Matrices(SQ, vdz, 3, 3, SSSS, 3);

hm = 0.569;

A\_1 = (double\*)malloc(3 \* 3 \* sizeof(double));

for(i = 0; i < 9; i++)

A\_1[i] = SQ[i] \* hm / hmb;

for(i = 0; i < 3; i++)

printf("%lf %lf %lf\r\n", A\_1[i\*3], A\_1[i\*3+1], A\_1[i\*3+2]);

free(D);

free(S);

free(C);

free(S11);

free(S12);

free(S12t);

free(S22);

free(S22\_1);

free(S22a);

free(S22b);

free(SS);

free(E);

free(U);

free(SSS);

free(eigen\_real);

free(eigen\_imag);

free(v1);

free(v2);

free(v);

free(Q);

free(Q\_1);

free(B);

free(QB);

free(SSSS);

free(eigen\_real3);

free(eigen\_imag3);

free(Dz);

free(vdz);

free(SQ);

free(A\_1);

return 0;

}

// Place here the source code of all the routines which have been forward-declared, available at

// [http://www.mymathlib.com/matrices/](http://www.google.com/url?q=http%3A%2F%2Fwww.mymathlib.com%2Fmatrices%2F&sa=D&sntz=1&usg=AOvVaw2heDPrcfuhCpq7ElW-_UxX)

# Ellipsoid Fitting

<https://www.mathworks.com/matlabcentral/fileexchange/23377-ellipsoid-fitting>

The code implemented a special case of the ellipsoid fitting technique proposed in the paper Least Squares Ellipsoid Specific Fitting, corresponding to case k= 4. Just put the two files in the same directory and run "testFit4". You can change the noise level to observe the fitting accuracy and efficiency of the algorithm. It works well when the underlying shape characterized by the data points is quite spherical. However, if this is not the case, some modification is required. For the technical detail of the fitting method, please refer to the above mentioned paper, which can be found from the following webpage:  
<http://www2.computer.org/portal/web/csdl/doi?doc=abs/proceedings/gmp/2004/2078/00/20780335abs.htm>

### Cite As

Q LI (2024). Ellipsoid Fitting (https://www.mathworks.com/matlabcentral/fileexchange/23377-ellipsoid-fitting), MATLAB Central File Exchange. Retrieved September 23, 2024.

