MITSUBISHI ELECTRIC

Programmable Controller

MELSEC iQ-R series

MELSEC iQ-R C Controller Module/C Intelligent Function Module Programming Manual (Data Analysis)

SAFETY PRECAUTIONS

(Read these precautions before using this product.)

Before using C Controller module and C intelligent function module, please read this manual and the relevant manuals carefully and pay full attention to safety to handle the product correctly.

CONDITIONS OF USE FOR THE PRODUCT

(1) Mitsubishi programmable controller ("the PRODUCT") shall be used in conditions;

i) where any problem, fault or failure occurring in the PRODUCT, if any, shall not lead to any major or serious accident; and

ii) where the backup and fail-safe function are systematically or automatically provided outside of the PRODUCT for the case of any problem, fault or failure occurring in the PRODUCT.

(2) The PRODUCT has been designed and manufactured for the purpose of being used in general industries.

MITSUBISHI SHALL HAVE NO RESPONSIBILITY OR LIABILITY (INCLUDING, BUT NOT LIMITED TO ANY AND ALL RESPONSIBILITY OR LIABILITY BASED ON CONTRACT, WARRANTY, TORT, PRODUCT LIABILITY) FOR ANY INJURY OR DEATH TO PERSONS OR LOSS OR DAMAGE TO PROPERTY CAUSED BY the PRODUCT THAT ARE OPERATED OR USED IN APPLICATION NOT INTENDED OR EXCLUDED BY INSTRUCTIONS, PRECAUTIONS, OR WARNING CONTAINED IN MITSUBISHI'S USER, INSTRUCTION AND/OR SAFETY MANUALS, TECHNICAL BULLETINS AND GUIDELINES FOR the PRODUCT.

("Prohibited Application")

Prohibited Applications include, but not limited to, the use of the PRODUCT in;

- Nuclear Power Plants and any other power plants operated by Power companies, and/or any other cases in which the public could be affected if any problem or fault occurs in the PRODUCT.
- Railway companies or Public service purposes, and/or any other cases in which establishment of a special quality assurance system is required by the Purchaser or End User.
- Aircraft or Aerospace, Medical applications, Train equipment, transport equipment such as Elevator and Escalator, Incineration and Fuel devices, Vehicles, Manned transportation, Equipment for Recreation and Amusement, and Safety devices, handling of Nuclear or Hazardous Materials or Chemicals, Mining and Drilling, and/or other applications where there is a significant risk of injury to the public or property.

Notwithstanding the above, restrictions Mitsubishi may in its sole discretion, authorize use of the PRODUCT in one or more of the Prohibited Applications, provided that the usage of the PRODUCT is limited only for the specific applications agreed to by Mitsubishi and provided further that no special quality assurance or fail-safe, redundant or other safety features which exceed the general specifications of the PRODUCTs are required. For details, please contact the Mitsubishi representative in your region.

CONSIDERATIONS FOR USE

Considerations for the Wind River Systems product

C Controller module and C intelligent function module, have an embedded real-time operating system, VxWorks, manufactured by Wind River Systems, Inc. in the United States. We, Mitsubishi, make no warranty for the Wind River Systems product and will not be liable for any problems and damages caused by the Wind River Systems product during use of C Controller module and C intelligent function module.

For the problems or specifications of the Wind River Systems product, refer to the corresponding manual or consult Wind River Systems, Inc.

Contact information is available on the following website.

- Wind River Systems, Inc.: www.windriver.com

INTRODUCTION

Thank you for purchasing the Mitsubishi Electric MELSEC iQ-R series programmable controllers.

This manual describes the functions required for programming.

Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the MELSEC iQ-R series programmable controller to handle the product correctly.

Please make sure that the end users read this manual.

CONTENTS

RELEVANT MANUALS

Manual name [manual number]	Description	Available form
MELSEC iQ-R C Controller Module/C Intelligent Function Module Programming Manual (Data Analysis) [SH-081756ENG] (this manual)	Explains the programming specifications and dedicated function library for analyzing the data of a C Controller module and a C intelligent function module.	e-Manual PDF
MELSEC iQ-R C Controller Module User's Manual (Startup) [SH-081367ENG]	Explains the performance specifications, procedure before operation, and troubleshooting of a C Controller module.	Print book e-Manual PDF
MELSEC iQ-R C Controller Module User's Manual (Application) [SH-081369ENG]	Explains the functions, devices, and parameters of a C Controller module.	Print book e-Manual PDF
MELSEC iQ-R C Intelligent Function Module User's Manual (Startup) [SH-081566ENG]	Explains the specifications, procedure before operation, wiring, and operation examples of a C intelligent function module.	Print book e-Manual PDF
MELSEC iQ-R C Intelligent Function Module User's Manual (Application) [SH-081567ENG]	Explains the functions, input/output signals, buffer memory, parameter setting, and troubleshooting of a C intelligent function module.	Print book e-Manual PDF
CW Workbench/CW-Sim Operating Manual [SH-081373ENG]	Explains the system configuration, specifications, functions, and troubleshooting of CW Workbench/CW-Sim.	e-Manual PDF

Point

e-Manual refers to the Mitsubishi Electric FA electronic book manuals that can be browsed using a dedicated tool.

e-Manual has the following features:

- Required information can be cross-searched in multiple manuals.
- Other manuals can be accessed from the links in the manual.
- Hardware specifications of each part can be found from the product figures.
- Pages that users often browse can be bookmarked.

TERMS

Unless otherwise specified, this manual uses the following terms.

Term	Description
C Controller module	A generic term for MELSEC iQ-R series C Controller modules.
C Controller module dedicated function	A dedicated function library offered by a C Controller module. It is used to control a C Controller module.
C intelligent function module	A generic term for MELSEC iQ-R series C intelligent function modules.
C intelligent function module dedicated function	A dedicated function library offered by a C intelligent function module. It is used to control a C intelligent function module.
CW Workbench	An abbreviation for a C Controller module and C intelligent function module engineering tool, CW Workbench.
Data analysis function	A dedicated function library offered by a C Controller module and a C intelligent function module. It is used for data analysis processing.
MELSEC data link function	A data link function library offered by a C Controller module. It is used to access another CPU module as a connection target via network or in a multiple CPU system.
MELSEC iQ-R series data link function	A dedicated function library offered by a C intelligent function module. It is used to access an own station or modules on the network.
Memory card	A generic term for SD memory cards and SDHC memory cards.
R12CCPU-V	An abbreviation for R12CCPU-V C Controller modules.
RD55UP06-V	An abbreviation for RD55UP06-V C intelligent function modules.
Statistical analysis function	A dedicated function library offered by a C Controller module and a C intelligent function module. It is used for statistical analysis processing.
VxWorks	A product name for a real-time operating system manufactured by Wind River Systems, Inc.

1 COMMON ITEMS

A user program is created by using the VxWorks standard API functions*1 and dedicated function library offered by a C Controller module and a C intelligent function module.
Create a use program in accordance with the specification of VxWorks, the operating system of C Controller module and C intelligent function module.

*1 For details on the VxWorks standard API functions, refer to the following programmer's guide supported.
VxWorks"KERNEL PROGRAMMER'S GUIDE"

Dedicated function library

A dedicated function library offered by a C Controller module and a C intelligent function module is as follows:
This manual explains data analysis functions and statistical analysis functions.

Item	Dedicated function library
• C Controller module	• C Controller module dedicated functions • MELSEC data link functions
• C intelligent function module	• C intelligent function module dedicated functions • MELSEC iQ-R series data link functions
• C Controller module • C intelligent function module	• Data analysis functions • Statistical analysis functions

Point

For the execution procedure of a user program, refer to the following manuals.
- MELSEC iQ-R C Controller Module User's Manual (Startup)
- MELSEC iQ-R C Intelligent Function Module User's Manual (Startup)

For the development environment of a user program, refer to the following manual.
- CW Workbench/CW-Sim Operating Manual

1.1 Header File

Include the following header file in a user program to use data analysis functions and statistical analysis functions of the dedicated function library.

Target module	Header file
C Controller module C intelligent function module	DANLFunc.h

Point

The header file is stored in a module to be used and CW Workbench.
- MELSEC iQ-R C Controller Module User's Manual (Application)
- MELSEC iQ-R C Intelligent Function Module User's Manual (Startup)
- CW Workbench/CW-Sim Operating Manual

1.2 Data Analysis Functions

Data analysis functions are library functions to perform data analysis processing on a C Controller module or a C intelligent function module.

Program processing

The following shows the user program processing using a data analysis function.

1. Start a task.
2. Check whether to change the operating conditions of a data analysis function.

To change the operating conditions, go to the procedure 3.
When not to change the operating conditions, go to the procedure 4.

3. Change the operating conditions of the data analysis function. (☞ Page 13 DANL_SetOpCondition)
4. Call a data analysis function and perform data analysis processing.
5. End the task.

1.3 Statistical Analysis Functions

Statistical analysis functions are library functions to perform statistical analysis processing on a C Controller module or a C intelligent function module.

Program processing

The following shows the user program processing using a statistical analysis function.

1. Start a task.
2. Call a statistical analysis function and perform statistical analysis processing.
3. End the task.

1.4 Considerations

This section shows the considerations for a user program which uses a data analysis function and a statistical analysis function.

CW-Sim and CW-Sim Standalone

Data analysis functions and statistical analysis functions of dedicated function libraries are not supported by CW-Sim and CW-Sim Standalone.

For the development environment of a user program, refer to the following manual.

📖CW Workbench/CW-Sim Operating Manual

Startup of a task performing floating-point operations

Always specify the VX_FP_TASK option for the third argument of taskSpawn when activating the following tasks.

- A task performing floating-point operations
- A task calling a function that returns floating-point value
- A task calling a function that takes floating-point value as an argument
- A task calling a data analysis function
- A task calling a statistical analysis function

Activating the above task without the VX_FP_TASK option specified may cause the operating system runaway.

For information on specifying the VX_FP_TASK option in a script file, refer to the following section.

- 📖MELSEC iQ-R C Controller Module User's Manual (Startup)
- 📖MELSEC iQ-R C Intelligent Function Module User's Manual (Startup)

For more details on the VX_FP_TASK option, refer to the manual for VxWorks.

1.5 Considerations on Interrupt Service Routine (ISR)

Data analysis functions and statistical analysis functions cannot be used in an interrupt service routine (ISR).

2 FUNCTION LIST

This chapter shows the lists of data analysis functions and statistical analysis functions.

2.1 Data Analysis Functions

The data analysis functions are as listed below.

Data analysis functions

(1): C Controller module, (2): C intelligent function module

Function name	Description	Firmware version		Reference
		(1)	(2)	
DANL_SetOpCondition	To set operating conditions for data analysis.	'07' or later	'05' or later	☞ Page 13 DANL_SetOpCondition
DANL_GetOpCondition	To acquire operating conditions for data analysis.	'07' or later	'05' or later	☞ Page 14 DANL_GetOpCondition
DANL_DigitalFilter	To perform digital filter operation for the specified wave.	'07' or later	'05' or later	☞ Page 15 DANL_DigitalFilter
DANL_EnvelopeCalculation	To calculate the envelope of the specified wave.	'09' or later	'06' or later	☞ Page 18 DANL_EnvelopeCalculation
DANL_FFTSpectrum	To perform spectrum calculation using fast Fourier transform (FFT) for the specified wave.	'07' or later	'05' or later	☞ Page 19 DANL_FFTSpectrum
DANL_FindCrossPoint	To calculate the number of cross points of the specified wave and a reference value for the number of cross points specified to the maximum number of cross points.	'07' or later	'05' or later	☞ Page 21 DANL_FindCrossPoint
DANL_Peak	To calculate the peak values (maximum and minimum) of the specified wave.	'09' or later	'06' or later	☞ Page 25 DANL_Peak
DANL_RMS	To calculate an RMS (root mean square) of the specified wave.	'09' or later	'06' or later	☞ Page 26 DANL_RMS
DANL_BoundCompareTest	To compare the specified wave and a check value to check an upper/lower limit.	'07' or later	'05' or later	☞ Page 27 DANL_BoundCompareTest
DANL_AryBoundCompareTest	To compare the specified wave and a check value to check an upper/lower limit of the wave.	'07' or later	'05' or later	☞ Page 29 DANL_AryBoundCompareTest

2.2 Statistical Analysis Functions

The statistical analysis functions are as listed below.

Statistical analysis functions

(1): C Controller module, (2): C intelligent function module

Function name	Description	Firmware version		Reference
		(1)	(2)	
DANL_LeastSquare	To calculate a coefficient and a constant of a polynomial, and a multiple correlation coefficient by using a least-squares method for the specified array.	'09' or later	'06' or later	☞ Page 31 DANL_LeastSquare
DANL_MovingAverage	To calculate a moving average of the specified array.	'09' or later	'06' or later	☞ Page 33 DANL_MovingAverage
DANL_StandardDeviation	To calculate a standard deviation of the specified array.	'09' or later	'06' or later	☞ Page 35 DANL_StandardDeviation
DANL_Variance	To calculate a variance of the specified array.	'09' or later	'06' or later	☞ Page 36 DANL_Variance
DANL_MTUnit	To determine a unit space that is used in the MT method based on the specified normal data.	'10' or later	'07' or later	☞ Page 37 DANL_MTUnit
DANL_MTMahalanobisDistance	To calculate a Mahalanobis distance of the specified input data.	'10' or later	'07' or later	☞ Page 41 DANL_MTMahalanobisDistance
DANL_MultipleRegression	To calculate a coefficient, constant, and regression statistics for multiple regression analysis.	'10' or later	'07' or later	☞ Page 44 DANL_MultipleRegression

3 DETAILS OF FUNCTIONS

This chapter shows the details of the data analysis functions and the statistical analysis functions.

3.1 Data Analysis Functions

DANL_SetOpCondition

This function sets the operating conditions of a data analysis function.

Format

short DANL_SetOpCondition (long* plSet, long lSetNum)

Argument

Argument	Name	Description	IN/OUT
plSet	Operating condition storage array	Specify operating conditions for data analysis. Prepare the array elements for the number of elements specified to the number of operating condition storage array elements (lSetNum).	IN
lSetNum	Number of operating condition storage array elements	Specify the number of array elements to set operating conditions. (Setting range: 2 or more)	IN

- The specification method of the operating condition storage array (plSet) is as follows:

Storage position	Default value	Description	Relevant function
plSet[0]	4	Specify the number of significant digits of a fractional value. The digits after the specified number of digits are rounded off. (Setting range: 0 to 14)	☞ Page 27 DANL_BoundCompareTest ☞ Page 29 DANL_AryBoundCompareTest
plSet[1]	1	Specify the number of consecutive points with which values are recognized as having exceeded or fallen below the reference value. The data is recognized as having exceeded or fallen below the reference value only when the number of cross points has exceeded or fallen below the reference value for the number of specified cross points consecutively. (Setting range: 1 to 100)	☞ Page 21 DANL_FindCrossPoint ☞ Page 27 DANL_BoundCompareTest ☞ Page 29 DANL_AryBoundCompareTest

Description

- The operating conditions of a data analysis function can be set by executing the DANL_SetOpCondition function.
- When the DANL_SetOpCondition function is not executed, the default values are applied to the operating conditions of the data analysis function.
- The values set with the DANL_SetOpCondition function are retained while the module is ON.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

Relevant functions

- Page 14 DANL_GetOpCondition
- Page 21 DANL_FindCrossPoint
- Page 27 DANL_BoundCompareTest
- Page 29 DANL_AryBoundCompareTest

DANL_GetOpCondition

This function acquires the operating conditions of a data analysis function.

Format

short DANL_GetOpCondition (long* plGet, long lGetNum)

Argument

Argument	Name	Description	IN/OUT
plGet	Operating condition storage array	The operating conditions of the data analysis function which are set in the module is stored. Prepare the array elements for the number of elements specified to the number of operating condition storage array elements (lGetNum).	OUT
lGetNum	Number of operating condition storage array elements	Specify the number of array elements to acquire operating conditions. (Setting range: 2 or more)	IN

Description

- Execute the DANL_GetOpCondition function to acquire the operating conditions of the data analysis function which are set in the module.
- The following information is stored to the operating condition storage array (plGet).

Storage position	Description
plGet[0]	The number of significant digits of a fractional value which is set in the module is stored. The digits after the stored number of digits are rounded off.
plGet[1]	The number of consecutive points with which values are recognized as having exceeded or fallen below the reference value set in the module is stored. The data is recognized as having exceeded or fallen below the reference value only when the number of cross points has exceeded or fallen below the reference value for the number of stored cross points consecutively.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

Relevant function

☞ Page 13 DANL_SetOpCondition

DANL_DigitalFilter

This function performs digital filter operation for the specified wave.

Format

short DANL_DigitalFilter (float* pfData, long lPoint, long lSamplingCycle, DIGITAL_FILTER_SETTINGS_STRUCT DigitalFilterSettings, float* pfOutput)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of wave data to be operated.	IN
lPoint	Data size	Specify the number of array elements of wave data to be operated. (Setting range: 16 to 1000000)	IN
lSamplingCycle	Sampling cycle	Specify the sampling cycle of the wave data to be operated. (Setting range: 1 to 1000000[us])	IN
DigitalFilterSettings	Digital filter setting	Specify the filter parameters set to the DIGITAL_FILTER_SETTINGS_STRUCT structure.	IN
pfOutput	Operation result storage destination	Specify the start address of a storage destination to store an operation result (array).	OUT

• The details of the DIGITAL_FILTER_SETTINGS_STRUCT structure are as follows.

No.	Type	Argument	Name	Description
1	long	lFilterType	Frequency response filter type	Specify a filter type. 0: Low-pass filter (LPF) 1: High-pass filter (HPF) 2: Band-pass filter (BPF) 3: Band elimination filter (BEF)
2	float	fCutoffFreqHz1	Cutoff frequency 1	Specify a cutoff frequency 1 [Hz]. Any value can be specified within the following range: • A value larger than or equal to 1.192093e-07 and less than or equal to the one obtained by dividing the sampling frequency by 2
3	float	fCutoffFreqHz2	Cutoff frequency 2	Specify a cutoff frequency 2 [Hz]. Any value can be specified within the following range: • A value larger than or equal to the one set to the cutoff frequency 1 and equal to or less than the one obtained by dividing the sampling frequency by 2 (Enabled only when BPF or BEF is specified to the cutoff frequency 2.)
4	long	lFilterCalcType	Digital filter type	Specify a filter type. (IIR filters can be used only when LPF or HPF is specified.) 0: FIR filter 1: IIR filter (Butterworth) 2: IIR filter (Chebyshev)
5	long	lDegree	Degree	Specify a degree. Any value can be specified within the following range: • For FIR filter: 2 to 200 (even number only) • For IIR filters: 2 to 40. A value less than or equal to the one specified to the data size (lPoint) should be specified.
6	float	fRipple	Ripple	Specify a ripple [dB]. Any value can be specified within the following range: • 0.015625 to 1.0 A ripple is used only when the IIR filter (Chebyshev) is specified.

Description

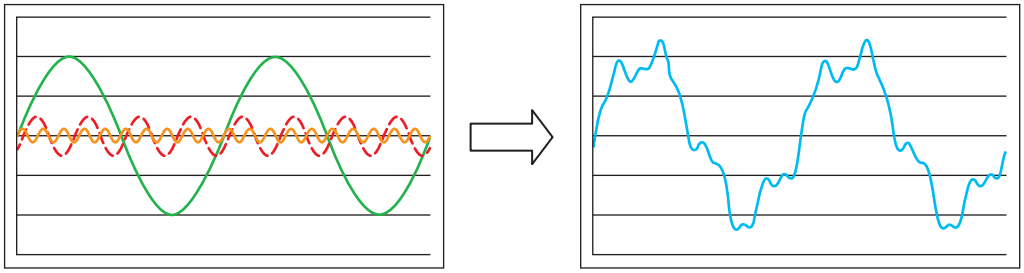
- A digital filter operation can be performed for the wave specified to the input data storage destination (pfData).
- Wave data for the digital filter operation is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- A result of the digital filter operation is stored in the operation result storage destination (pfOutput) for the size specified to the data size (lPoint) of the input data storage destination (pfData). Reserve an area larger than the size specified to the data size (lPoint) for the operation result storage destination (pfOutput).
- The DANL_DigitalFilter function supports low-pass filter, high-pass filter, band-pass filter, and band elimination filter.
- The DANL_DigitalFilter function supports FIR filter, IIR filter (Butterworth), and IIR filter (Chebyshev).

Operation example of the frequency response filter

The following shows an example when each filter is applied to the synthetic wave (2) which consists of three waves shown in the composed wave (1).

In the following figure, the horizontal axis indicates time and the vertical axis indicates amplitude. *1

*1 The following is an example of an operation. The actual results may vary.

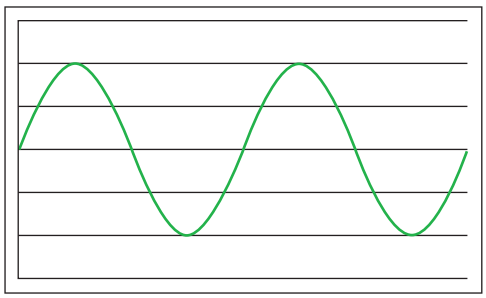


(1) (2)

Filter type	Feature
Low-pass filter	Attenuates and terminates signals with frequencies higher than the specified frequency to pass only low frequency signals.
High-pass filter	Attenuates and terminates signals with frequencies lower than the specified frequency to pass only high frequency signals.
Band-pass filter	Passes only signals with frequencies within the specified range.
Band elimination filter	Eliminates frequency signals within the specified range.

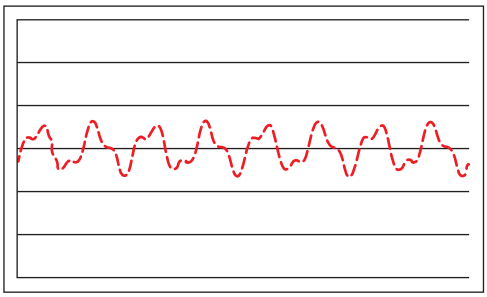
● Low-pass filter

A low-pass filter attenuates and terminates signals with a frequency higher than the one specified to the cutoff frequency 1 (fCutoffFreqHz1) to pass only low frequency signals.



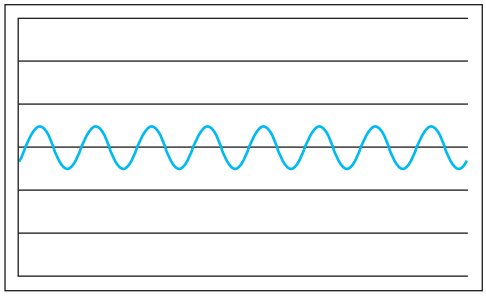
● High-pass filter

A high-pass filter attenuates and terminates signals with a frequency lower than the one specified to the cutoff frequency 1 (fCutoffFreqHz1) to pass only high frequency signals.



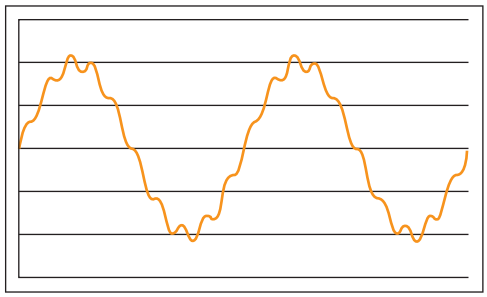
● Band-pass filter

A band-pass filter passes signals with a frequency higher than the one specified to the cutoff frequency 1 (fCutoffFreqHz1) and lower than the one specified to the cutoff frequency 2 (fCutoffFreqHz2).



● Band elimination filter

A band elimination filter eliminates signals with a frequency higher than the one specified to the cutoff frequency 1 (fCutoffFreqHz1) and lower than the one specified to the cutoff frequency 2 (fCutoffFreqHz2).



Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

DANL_EnvelopeCalculation

This function calculates the envelope of the specified wave.

Format

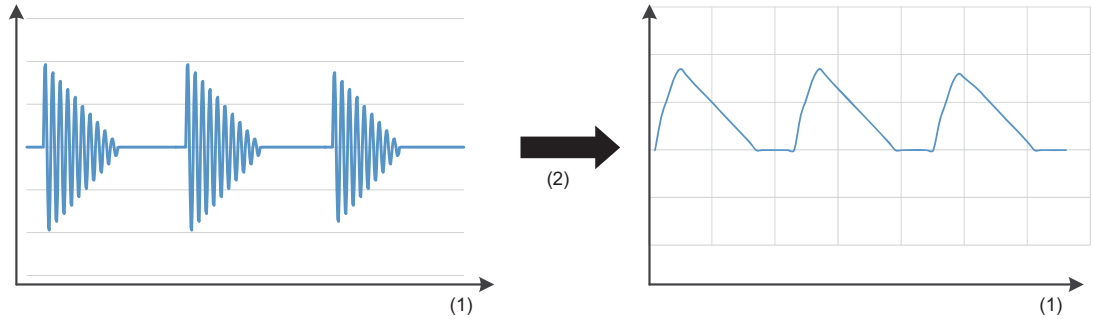
short DANL_EnvelopeCalculation (float* pfData, long lPoint, float* pfOutput)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of wave data to be operated.	IN
lPoint	Data size	Specify the number of array elements of wave data to be operated. (Setting size: 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768)	IN
pfOutput	Operation result storage destination	Specify the start address of a storage destination to store an operation result (array).	OUT

Description

- The envelope of a wave specified to the input data storage destination (pfData) can be calculated.



(1): Time
(2): Envelope calculation

- Wave data for the envelope calculation is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- An operation result is stored in the operation result storage destination (pfOutput) for the size specified to the data size (lPoint) of the input data storage destination (pfData). Reserve an area larger than the size specified to the data size (lPoint) for the operation result storage destination (pfOutput).
- If an overflow occurs during operation, 'Inf' or 'NaN' is stored in the operation result storage destination (pfOutput). Review the input data.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

DANL_FFTSpectrum

This function performs spectrum calculation using fast Fourier transform (FFT) for the specified wave.

Format

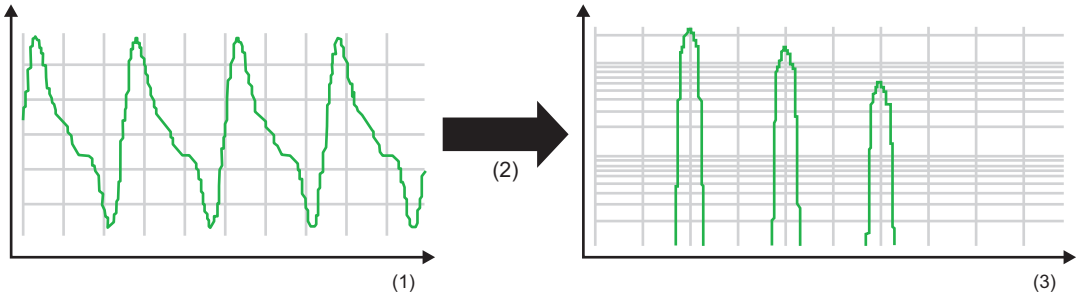
short DANL_FFTSpectrum (float* pfData, long lPoint, long lWindowType, long lSpectrumFormat, float* pfOutput)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of wave data to be operated.	IN
lPoint	Data size	Specify the number of array elements of wave data to be operated. (Setting range: 16 to 32768)	IN
lWindowType	Window type	Specify a window function. 1: Rectangular 2: Hanning 3: Hamming 4: Blackman	IN
lSpectrumFormat	Spectrum format	Specify an output spectrum format. 1: Power 2: P (Peak) 3: P-P (Peak to Peak) 4: RMS (Root Mean Square)	IN
pfOutput	Operation result storage destination	Specify the start address of a storage destination to store an operation result (array).	OUT

Description

- The fast Fourier transform (FFT) is performed for the wave data specified to the input data storage destination (pfData) to perform spectrum calculation. When the data size (lPoint) of the specified wave is other than the power of 2, a discrete Fourier transform (DFT) is applied, and as a result, the execution speed is slower than fast Fourier transform (FFT).
- By performing the fast Fourier transform (FFT), the time axis of a wave is converted to a frequency axis.



(1): Time
(2): FFT operation
(3): Frequency

- Wave data for the fast Fourier transform (FFT) analysis is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- An operation result of the fast Fourier transform (FFT) is stored in the operation result storage destination (pfOutput) for the size specified to the data size (lPoint) of the input data storage destination (pfData). Reserve an area larger than the size specified to the data size (lPoint) for the operation result storage destination (pfOutput).
- The DANL_FFTSpectrum function supports window functions (rectangular, hanning, hamming, and blackman).
- The DANL_FFTSpectrum function supports output spectrum formats (power, P (Peak), P-P (Peak to Peak), and RMS (Root Mean Square)).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

DANL_FindCrossPoint

This function calculates the number of cross points of the specified wave and a reference value for the number of cross points specified to the maximum number of cross points.

Format

short DANL_FindCrossPoint (float* pfData, long lPoint, float fReferenceValue, long lCrossPattern, long lMaxPoint, long* plCrossData,long* plCrossPoint)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of wave data to be operated.	IN
lPoint	Data size	Specify the number of array elements of wave data to be operated. (Setting range: 1 to 1000000)	IN
fReferenceValue	Reference value	Specify the reference value to be operated.	IN
lCrossPattern	Cross point recognition pattern	Specify the pattern to be recognized as a cross point. 0: Both rise and fall 1: Rise only 2: Fall only	IN
lMaxPoint	Maximum number of cross points	Specify the maximum number of cross points to be recognized. (Setting range: -1, 1 to data size (lPoint)) When -1 is specified, the data is not stored to the cross point index storage destination (plCrossData). Instead, the number of recognized cross points is only counted.	IN
plCrossData	Cross point index storage destination	The indexes (1-Point or array) of the cross points are stored until the number of cross points reaches the maximum number of cross points (lMaxPoint). When the maximum number of cross points (lMaxPoint) is -1, the value of an array does not change.	OUT
plCrossPoint	Number of cross points	The number of recognized cross points (1-point) is stored.	OUT

- Optional setting (An operating condition can be changed with the DANL_SetOpCondition function.)

Setting	Default value	Description
Number of times of reference value determinations (Set to the second argument: plSet[1])	1	Specify the number of consecutive points with which values are recognized as having exceeded or fallen below the reference value. The data is recognized as having exceeded or fallen below the reference value only when the number of cross points has exceeded or fallen below the reference value for the number of specified cross points consecutively. (Setting range in the DANL_SetOpCondition function: 1 to 100)

Description

- The number of cross points is calculated using the wave specified to the input data storage destination (pfData) and the value specified to the reference value (fReferenceValue), and the result is stored to the cross point (plCrossPoint).
- A cross point is recognized only when the number of elements in the input data storage destination (pfData) has exceeded or fallen below the reference value (fReferenceValue) consecutively for the number of times specified to the number of times of the reference value determinations in the optional setting.
- Wave data for calculating cross points is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- The index of a cross point is stored in the cross point index storage destination (plCrossData). Reserve an area larger than the value specified to the number of maximum cross points (lMaxPoint) for the cross point index storage destination (plCrossData).
- The DANL_FindCrossPoint function stops processing when the number of cross points reaches the maximum number of cross points (lMaxPoint).
- If no cross points exist, '0' will be stored to the cross point (plCrossPoint). The indexes in the cross point index storage destination (plCrossData) do not change.

Recognition pattern of a cross point

The index of the next cross point is stored to the cross point index storage destination (plCrossData). When the specified wave and the reference value cross between pfData[k] and pfData[k+1] (integer within the range of k≥0), the index of the cross point to be stored to the cross point index storage destination (plCrossData) is (k+1). The following shows the examples of patterns that recognizes cross points.

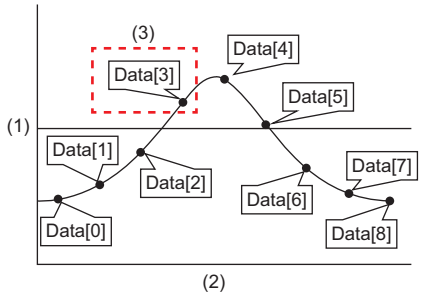
● Rise only

A rise is detected when '3' or less is set to the number of times of reference value determinations in the DANL_SetOpCondition function.

Number of cross points: 1 point

Crossing position: Data[3]

Index of a cross point: 3



(1): Reference value
(2): Time
(3): Crossing position

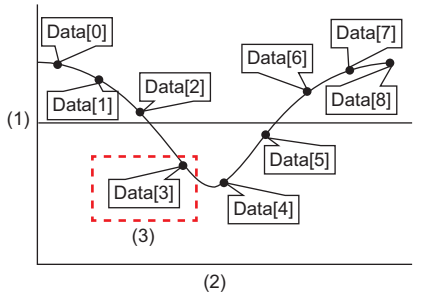
● Fall only

A fall is detected when '3' or less is set to the number of times of reference value determinations in the DANL_SetOpCondition function.

Number of cross points: 1 point

Crossing position: Data[3]

Index of a cross point: 3



(1): Reference value
(2): Time
(3): Crossing position

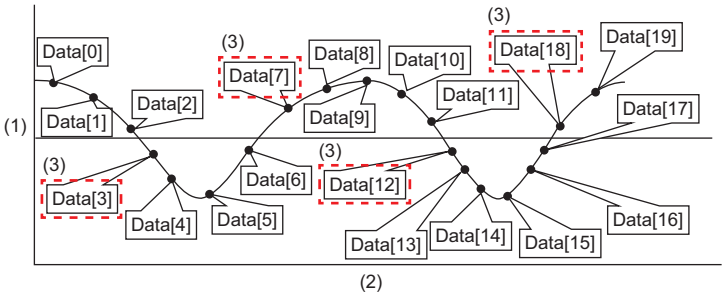
● Both rise and fall

The total of 4 cross points are detected when '2' or less is set to the number of times of reference value determinations with the DANL_SetOpCondition function.

Number of cross points: 4 points

Crossing positions: Data[3], Data[7], Data[12], Data[18]

Indexes of cross points: 3, 7, 12, 18



(1): Reference value
(2): Time
(3): Crossing position

- After data exceeded or fell below (rise/fall) the reference value for the number of times of reference value determinations set to plSet[1] in the DANL_SetOpCondition function, if the data did not fall below or exceed (fall/rise) the reference value consecutively, then the next fall/rise will not be detected.
- Even when only the number of units of data less than the number of times of reference value determinations exists before a first cross point, it is recognized as a first cross point when the number of units of data more than the number of times of reference value determinations exists after the cross point. As for a last cross point, if the number of units of data more than the number of times of reference value determinations does not exist after a last cross point, it will not be recognized as a last cross point.
- Regarding the figure in "Both rise and fall" above, the following table shows how cross points are recognized when each value (1 to 6, 7 or more) is set to the number of times of reference value determinations.

Number of times of reference value determinations	Description
1 to 2	'3', '7', '12', and '18', are stored to the cross point storage destination (plCrossData). ('3' is stored to plCrossData[0], '7' is stored to plCrossData[1], '12' is stored to plCrossData[2], and '18' is stored to plCrossData[3].)
3 to 4	'3', '7', and '12', are stored to the cross point storage destination (plCrossData). ('3' is stored to plCrossData[0], '7' is stored to plCrossData[1], and '12' is stored to plCrossData[2].) • Only three points' worth of data (Data[0] to Data[2]) exist before Data[3], however, Data[3] is recognized as a cross point (fall) because the first cross point is recognized as a cross point when the number of units of data more than the specified number of times of reference value determinations exist after the first cross point. Additionally, since the number of units of data more than the specified number of times of reference value determinations does not exist after the last cross point, Data[18] is not recognized as a cross point (rise).
5 to 6	'12' is stored to the cross point index storage destination (plCrossData). ('12' is stored to plCrossData[0].) • Since only four points' worth of data (Data[3] to Data[6]) exist after Data[3], Data[3] is not recognized as a cross point (fall); therefore, Data[3] to Data[6] are not recognized as being below (having fallen below) the reference value, and as a result, Data[0] to Data[6] are recognized as being above the reference value. Additionally, five points' worth of data (Data[7] to Data[11]) exist after Data[7]; however, Data[7] is not recognized as a cross point (rise) due to the reason given above. Consequently, 12 points' worth of data (Data[0] to Data[11]) before Data[12] is recognized as being above the reference value.
7 or more	The data is recognized as having no cross points.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

Relevant function

- Page 13 DANL_SetOpCondition

DANL_Peak

This function calculates the peak values (maximum and minimum) of the specified wave.

Format

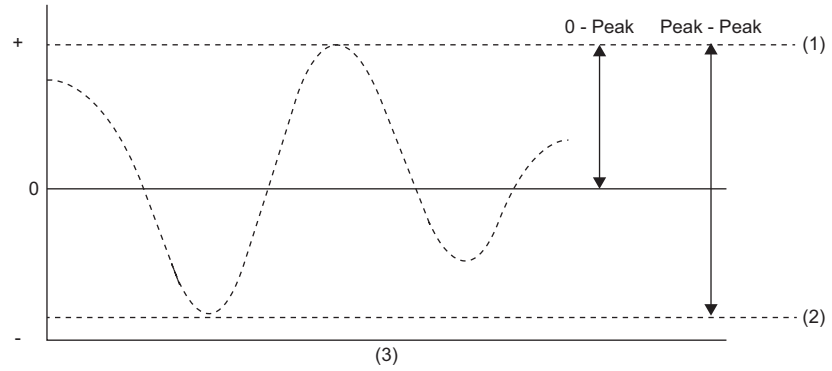
short DANL_Peak (float* pfData, long lPoint, float* pfMaxData, float* pfMinData)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of wave data to be determined.	IN
lPoint	Data size	Specify the number of array elements of wave data to be operated. (Setting range: 1 to 1000000)	IN
pfMaxData	Maximum value	The maximum value (1-point) is stored.	OUT
pfMinData	Minimum value	The minimum value (1-point) is stored.	OUT

Description

- The maximum value and the minimum value of a wave specified to the input data storage destination (pfData) can be calculated.
- Wave data for the peak value (maximum and minimum) calculation is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- The following peak values can be calculated by the maximum value and the minimum value calculated with the DANL_Peak function.
  P (0 - Peak): Larger one of the |maximum value| or the |minimum value|
  P-P (Peak - Peak): Values from the maximum value to the minimum value



(1): Maximum value
(2): Minimum value
(3): Time

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

DANL_RMS

This function calculates an RMS (root mean square) of the specified wave.

Format

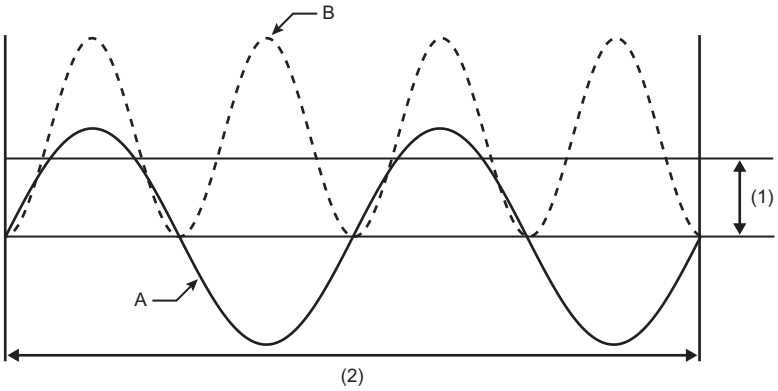
short DANL_RMS (float* pfData, long lPoint, float* pfOutput)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of wave data to be determined.	IN
lPoint	Data size	Specify the number of array elements of wave data to be operated. (Setting range: 1 to 1000000)	IN
pfOutput	Operation result	An operation result (1-point) is stored.	OUT

Description

- An RMS (root mean square) of a wave specified to the input data storage destination (pfData) can be calculated.
- The magnitude of time-varying signal is indicated, and the degree of dispersion of data is applied compared with a normal average value.
- The square root of a value obtained by dividing the sum of square values of input data by a value specified to the data size (lPoint) is stored in the operation result (*pfOutput) as an RMS (root mean square).



A: Input data
B: Square value of input data
(1): RMS (root mean square)
(2): Data size (lPoint)

- Wave data for the RMS (root mean square) calculation is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- An operation result of the DANL_RMS function can be obtained by the following formula. An RMS (root mean square) can be calculated for one unit of wave data (sampled data for the size specified to the data size (lPoint) {pfData[0], pfData[1], ..., pfData[lPoint-1]}).

$$*pfOutput = \sqrt{\frac{1}{lPoint}\sum_{i=0}^{lPoint-1} pfData[i]^2} = \sqrt{\frac{pfData[0]^2 + pfData[1]^2 + \cdots + pfData[lPoint-1]^2}{lPoint}}$$

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

DANL_BoundCompareTest

This function compares the specified wave and a check value to check an upper/lower limit.

Format

short DANL_BoundCompareTest (float* pfData, long lPoint, float fLowerLimit, float fUpperLimit, long* plResult)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of wave data to be determined.	IN
lPoint	Data size	Specify the number of array elements of wave data to be operated. (Setting range: 1 to 1000000)	IN
fLowerLimit	Lower limit value	Specify a check value (lower limit value).	IN
fUpperLimit	Upper limit value	Specify a check value (upper limit value).	IN
plResult	Check result	A check result (1-point) is stored. 0: OK 1: NG	OUT

- Optional setting (An operating condition can be changed with the DANL_SetOpCondition function.)

Setting	Default value	Description
Number of significant digits (Set to the first argument: plSet[0])	4	Specify the number of significant digits of a fractional value. The digits after the specified number of digits are rounded off. (Setting range in the DANL_SetOpCondition function: 0 to 14)
Number of times of reference value determinations (Set to the second argument: plSet[1])	1	Specify the number of consecutive points with which values are recognized as having exceeded or fallen below the reference value. The data is recognized as having exceeded or fallen below the reference value only when the number of cross points has exceeded or fallen below the reference value for the number of specified cross points consecutively. (Setting range in the DANL_SetOpCondition function: 1 to 100)

Description

- The wave specified to the input data storage destination (pfData) and the check value (upper/lower limit value) are compared. The result will be NG if the element of the input data storage destination (pfData) exceeds or falls below the check value (upper/lower limit value). However, when the specified wave is matched with the check value (upper/lower limit value), the result is OK.
  When the operation conditions are not changed with the DANL_SetOpCondition function, the default value (1) is set to the number of times of reference value determinations. At this time, if the value exceeds/falls below the check value (upper/lower limit value) even once, the result will be NG.
- The values in the input data storage destination (pfData), lower limit value (fLowerLimit), and upper limit value (fUpperLimit) are rounded off to the number of digits less than or equal to the one specified for the optional setting.

Ex.

When the number of significant digits is '4' (default), each setting value is rounded off as follows:
Input data storage destination (pfData) is 1.123489: 1.1235
Lower limit value (fLowerLimit) is 0.123489: 0.1235
Upper limit value (fUpperLimit) is 2.123489: 2.1235

- Specified wave data is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).

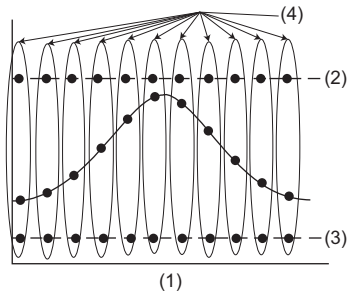
Effect of rounding error

Setting the number of significant digits of a fractional value does not completely prevent the occurrence of rounding errors. For example, '1.1115' is expressed as '1.1114999...' in the module which handles three-digit significant digits. Therefore, if '1.1115' is rounded off to three decimal places, the result will be '1.111', not '1.112'.
To avoid the effect of rounding errors, convert the value to integer.

Operation example for checking an upper/lower limit

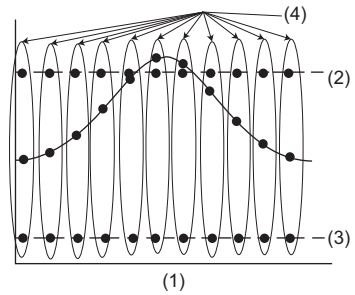
The operation example of the DANL_BoundCompareTest function is as follow:

- By executing the DANL_BoundCompareTest function, OK (0) is stored to the check result (plResult) regardless of the values set with the DANL_SetOpCondition function.



(1): Time
(2): Upper limit value
(3): Lower limit value
(4): Compare each nth data. The result is OK only when the value is within the range of 'Lower limit value ≤ specified wave ≤ upper limit value'.

- Depending on the number of times of reference value determinations set with the DANL_SetOpCondition function, the value to be stored to the check result (plResult) differs. NG (1) is stored when 2 or less is set, and OK (0) is stored when 3 or more is set.



(1): Time
(2): Upper limit value
(3): Lower limit value
(4): Compare each nth data. The result is OK only when the value is within the range of 'Lower limit value ≤ specified wave ≤ upper limit value'.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

Relevant function

- Page 13 DANL_SetOpCondition

DANL_AryBoundCompareTest

This function compares the specified wave and a check value to check an upper/lower limit of the wave.

Format

short DANL_AryBoundCompareTest (float* pfData, long lPoint, float* pfLowerLimit, float* pfUpperLimit, long* plResult)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of wave data to be determined.	IN
lPoint	Data size	Specify the number of array elements of wave data to be operated. (Setting range: 1 to 1000000)	IN
pfLowerLimit	Lower limit wave storage destination	Specify the start address where a check value (lower limit wave) is to be stored.	IN
pfUpperLimit	Upper limit wave storage destination	Specify the start address where a check value (upper limit wave) is to be stored.	IN
plResult	Check result	A check result (1-point) is stored. 0: OK 1: NG	OUT

- Optional setting (An operating condition can be changed with the DANL_SetOpCondition function.)

Setting	Default value	Description
Number of significant digits (Set to the first argument: plSet[0])	4	Specify the number of significant digits of a fractional value. The digits after the specified number of digits are rounded off. (Setting range in the DANL_SetOpCondition function: 0 to 14)
Number of times of reference value determinations (Set to the second argument: plSet[1])	1	Specify the number of consecutive points with which values are recognized as having exceeded or fallen below the reference value. The data is recognized as having exceeded or fallen below the reference value only when the number of cross points has exceeded or fallen below the reference value for the number of specified cross points consecutively. (Setting range in the DANL_SetOpCondition function: 1 to 100)

Description

- The wave specified to the input data storage destination (pfData) and the check value (upper/lower limit wave) are compared. The result will be NG if the nth data of the input data storage destination (pfData) exceeds or falls below the nth data of the check value (upper/lower limit wave). However, when the specified wave is matched with the check value (upper/lower limit wave), the result is OK.
  When the operation conditions are not changed with the DANL_SetOpCondition function, the default value (1) is set to the number of times of reference value determinations. At this time, if the value exceeds/falls below the check value (upper/lower limit wave) even once, the result will be NG.
- Specified wave data is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- A check value (upper or lower limit wave) is read from the start address specified to the upper limit wave storage destination (pfUpperLimit) or the lower limit wave storage destination (pfLowerLimit) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the upper limit wave storage destination (pfUpperLimit) or the lower limit wave storage destination (pfLowerLimit).
- The values in the input data storage destination (pfData), lower limit wave storage destination (pfLowerLimit), and upper limit wave storage destination (pfUpperLimit) are rounded off to the number of digits less than or equal to the one specified for the optional setting.

Ex.

(Example) When the number of significant digits is '4' (default), each setting value is rounded off as follows:
Input data storage destination (pfData) is 1.123489: 1.1235
Lower limit wave storage destination (pfLowerLimit) is 0.123489: 0.1235
Upper limit wave storage destination (pfUpperLimit) is 2.123489: 2.1235

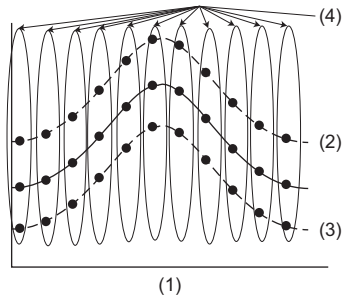
Effect of rounding error

Setting the number of significant digits of a fractional value does not completely prevent the occurrence of rounding errors. For example, '1.1115' is expressed as '1.1114999...' in the module which handles three-digit significant digits. Therefore, if '1.1115' is rounded off to three decimal places, the result will be '1.111', not '1.112'.
To avoid the effect of rounding errors, convert the value to integer.

Operation example for checking an upper/lower limit of a wave

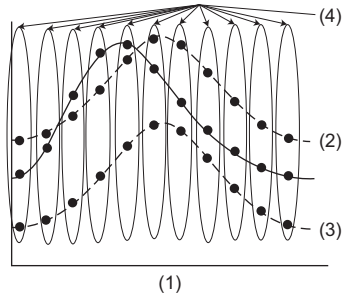
The operation example of the DANL_AryBoundCompareTest function is as follow:

- By executing the DANL_AryBoundCompareTest function, OK (0) is stored to the check result (plResult) regardless of the values set with the DANL_SetOpCondition function.



(1): Time
(2): Upper limit wave
(3): Lower limit wave
(4): Compare each nth data. The result is OK only when the value is within the range of 'Lower limit value ≤ specified wave ≤ upper limit value'.

- Depending on the number of times of reference value determinations set with the DANL_SetOpCondition function, the value to be stored to the check result (plResult) differs. NG (1) is stored when 3 or less is set, and OK (0) is stored when 4 or more is set.



(1): Time
(2): Upper limit wave
(3): Lower limit wave
(4): Compare each nth data. The result is OK only when the value is within the range of 'Lower limit value ≤ specified wave ≤ upper limit value'.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 48 Data analysis functions

Relevant function

- Page 13 DANL_SetOpCondition

3.2 Statistical Analysis Functions

DANL_LeastSquare

This function calculates a coefficient and a constant of a polynomial, and a multiple correlation coefficient using a least-squares method for the specified array.

Format

short DANL_LeastSquare (float* pfDataX, float* pfDataY, long lPoint, long lDegree, float* pfOutput, float* pfCoefficient)

Argument

Argument	Name	Description	IN/OUT
pfDataX	X coordinate storage destination	Specify the start address of an X coordinate array to be calculated.	IN
pfDataY	Y coordinate storage destination	Specify the start address of a Y coordinate array to be calculated.	IN
lPoint	Data size	Specify the number of array elements of the X coordinate and the Y coordinate to be calculated. (Setting range: 3 to 1000000)*1	IN
lDegree	Degree	Specify a degree to be operated. (Setting range: 1 to 10)	IN
pfOutput	Operation result storage destination	Specify the start address of a storage destination to store an operation result (array).	OUT
pfCoefficient	Multiple correlation coefficient	A multiple correlation coefficient (1-point) is stored.	OUT

*1 Specify a value larger than or equal to the one specified to the degree (lDegree) + 2.

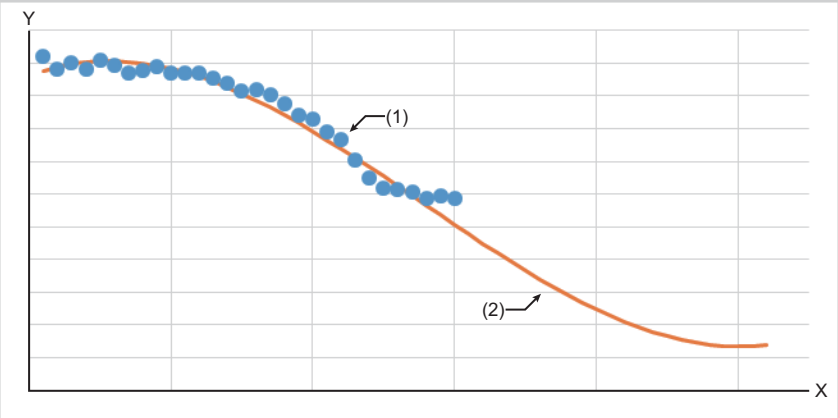
Description

- A least-squares method operation can be performed for an array (X axis and Y axis) specified to the X coordinate storage destination (pfDataX) and the Y coordinate storage destination (pfDataY).
- An array for a least-squares method calculation is read from the start addresses specified to the X coordinate storage destination (pfDataX) and the Y coordinate storage destination (pfDataY) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the X coordinate storage destination (pfDataX) and the Y coordinate storage destination (pfDataY).
- An operation result is stored in the operation result storage destination (pfOutput). The number of arrays in the operation result storage destination (pfOutput) will be 'degree + 1'. Reserve an area equal to or larger than the value specified to the degree (lDegree) + 1 for the operation result storage destination (pfOutput).
- If an overflow occurs during operation, 'Inf' or 'NaN' is stored in the operation result storage destination (pfOutput) or the multiple correlation coefficient (pfCoefficient). Review the input data.

- For an operation result, each coefficient is stored from the start in the order of {constant, first-order coefficient, second-order coefficient, third-order coefficient...} for the degree specified to the degree (lDegree).
  Example: When '5' is specified to the degree (lDegree), the number of arrays in the operation result storage destination (pfOutput) will be '6'.
  (Coefficients (a to e) and a constant (f) of a fifth-order polynomial are calculated.)
  Prediction data Y to the X coordinate can be calculated by using the operation result obtained above and by specifying an arbitrary value to 'X' in the polynomial '$Y = aX^5 + bX^4 + cX^3 + dX^2 + eX + f$'.

Number of arrays in the operation result storage destination (pfOutput)	Data item	Corresponding coefficient
pfOutput[0]	Constant	f
pfOutput[1]	First-order coefficient	e
pfOutput[2]	Second-order coefficient	d
pfOutput[3]	Third-order coefficient	c
pfOutput[4]	Fourth-order coefficient	b
pfOutput[5]	Fifth-order coefficient	a

Prediction data Y to the X coordinate



X: X axis
Y: Y axis
(1): Input data
(2): Prediction data Y

- A multiple correlation coefficient has the following characteristics:
  '-1 <= multiple correlation coefficient <= 1'
  Close to 1: Positive correlation (with a straight line having a positive slope)
  Close to 0: No correlation
  Close to -1: Negative correlation (with a straight line having a negative slope)

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 50 Statistical analysis function

DANL_MovingAverage

This function calculates a moving average of the specified array.

Format

short DANL_MovingAverage (float* pfData, long lPoint, long lAveragePoint, float* pfOutput)

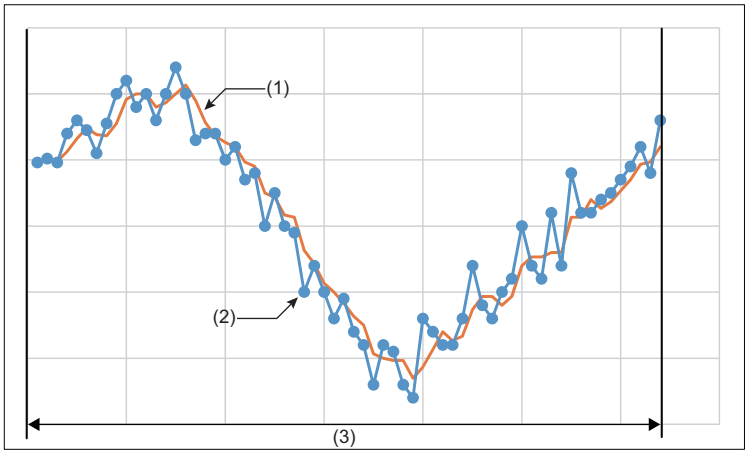
Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of an array to be determined.	IN
lPoint	Data size	Specify the number of array elements to be operated. (Setting range: 2 to 1000000)	IN
lAveragePoint	Number of moving average points	Specify the number of moving average points.*1 (Setting range: 1 to 1000)	IN
pfOutput	Operation result storage destination	Specify the start address of a storage destination to store an operation result (array).	OUT

*1 Specify a value smaller than one specified to the data size (lPoint).

Description

- A moving average of an array specified to the input data storage destination (pfData) can be calculated.



(1): Moving average
(2): Input data
(3): Data size (lPoint)

- An array for the moving average calculation is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- An operation result is stored in the operation result storage destination (pfOutput) for the size specified to the data size (lPoint) of the input data storage destination (pfData). Reserve an area larger than the size specified to the data size (lPoint) for the operation result storage destination (pfOutput).
- An operation result of the DANL_MovingAverage function can be obtained by the following formula. (Average of the number of moving average points (lAveragePoint) up to pfData[i])

$$pfOutput[i] = \frac{\{pfData[i] + pfData[i - 1] + \cdots + pfData[i - lAveragePoint + 1]\}}{lAveragePoint}$$

- When executing 'DANL_MovingAverage(pfData,10,3,pfOutput);' by specifying the following values to each input data storage destination (pfData), '0' (normal) is returned and the following values are stored in each operation result storage destination (pfOutput).

Value stored in the input data storage destination (pfData)

pfData[0]	pfData[1]	pfData[2]	pfData[3]	pfData[4]	pfData[5]	pfData[6]	pfData[7]	pfData[8]	pfData[9]
10.0	20.0	30.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0

Value stored in the operation result storage destination (pfOutput)

pfOutput[0]	pfOutput[1]	pfOutput[2]	pfOutput[3]	pfOutput[4]
10.0/1	(20.0+10.0)/2	(30.0+20.0+10.0)/3	(40.0+30.0+20.0)/3	(50.0+40.0+30.0)/3

pfOutput[5]	pfOutput[6]	pfOutput[7]	pfOutput[8]	pfOutput[9]
(60.0+50.0+40.0)/3	(70.0+60.0+50.0)/3	(80.0+70.0+60.0)/3	(90.0+80.0+70.0)/3	(100.0+90.0+80.0)/3

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 50 Statistical analysis function

DANL_StandardDeviation

This function calculates a standard deviation of the specified array.

Format

short DANL_StandardDeviation (float* pfData, long lPoint, float* pfOutput)

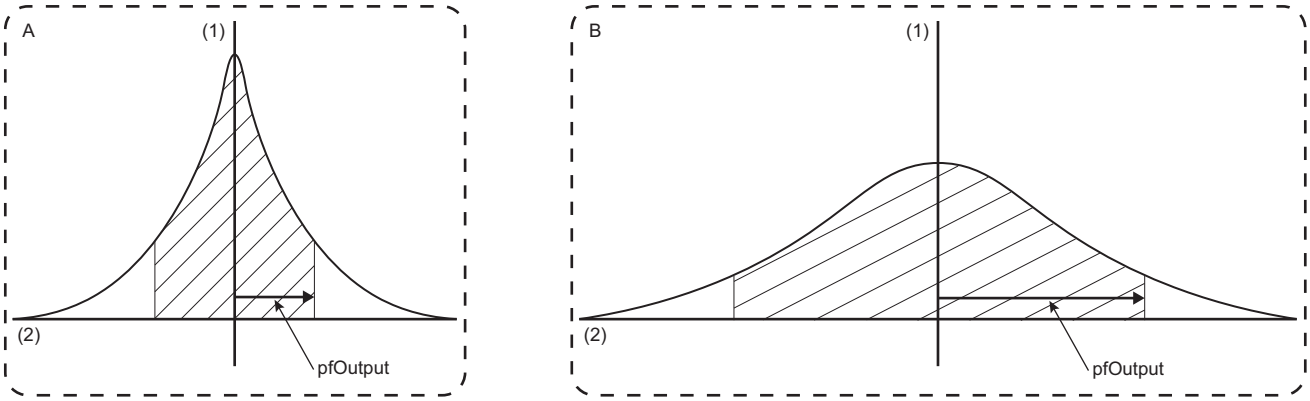
Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of an array to be determined.	IN
lPoint	Data size	Specify the number of array elements to be operated. (Setting range: 1 to 1000000)	IN
pfOutput	Operation result	An operation result (1-point) is stored.	OUT

Description

- A standard deviation of an array specified to the input data storage destination (pfData) can be calculated.

Average value of input data



A: Small standard deviation = Small dispersion of input data
B: Large standard deviation = Large dispersion of input data
(1): Probability density
(2): Random variable

- An array for the standard deviation calculation is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- An operation result of the DANL_StandardDeviation function can be obtained by the following formula.

$$*pfOutput = \sqrt{\frac{1}{lPoint}\sum_{i=0}^{lPoint-1}\left[pfData[i] - \underbrace{\left[\frac{1}{lPoint}\sum_{j=0}^{lPoint-1}pfData[j]\right]}_{(1)}\right]^2}$$

(1): Average value of input data

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 50 Statistical analysis function

DANL_Variance

This function calculates a variance of the specified array.

Format

short DANL_Variance (float* pfData, long lPoint, float* pfOutput)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of an array to be determined.	IN
lPoint	Data size	Specify the number of array elements to be operated. (Setting range: 1 to 1000000)	IN
pfOutput	Operation result	An operation result (1-point) is stored.	OUT

Description

- A variance of the array specified to the input data storage destination (pfData) is obtained. (The square root of a variance is a standard deviation.)
- An array for the variance calculation is read from the start address specified to the input data storage destination (pfData) for the size specified to the data size (lPoint). Reserve an area larger than the size specified to the data size (lPoint) for the input data storage destination (pfData).
- If an overflow occurs during operation, 'Inf' or 'NaN' is stored in the operation result (pfOutput). Review the input data.
- An operation result of the DANL_Variance function can be obtained by the following formula.

$$*\mathrm{pfOutput} = \frac{1}{\mathrm{lPoint}} \sum_{i=0}^{\mathrm{lPoint}-1} \left[\mathrm{pfData}[i] - \underbrace{\left[\frac{1}{\mathrm{lPoint}} \sum_{j=0}^{\mathrm{lPoint}-1} \mathrm{pfData}[j] \right]}_{(1)} \right]^2$$

(1): Average value of input data

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 50 Statistical analysis function

DANL_MTUnit

This function determines a unit space that is used in the MT method based on the specified normal data.

Format

short DANL_MTUnit(float* pfData, long lDataSize, long lItemNum, long lSampleNum, long lCorrelation, long lSN, MTUNIT_MDDATA_STRUCT* pMTUnitMDData, MTUNIT_OUTPUT_STRUCT* pMTUnitOutput)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of an array to be determined.	IN
lDataSize	Input data storage destination size (byte)	Specify a size of the array to be determined. Specify a value greater than or equal to the one specified to the number of normal data items (lItemNum) × number of normal data samples (lSampleNum) × number of bytes required for float type.	IN
lItemNum	Number of normal data items	Specify the number of normal data items. Specify a value less than or equal to the one specified to the number of normal data samples (lSampleNum). (Setting range: 2 to 300)	IN
lSampleNum	Number of normal data samples	Specify the number of normal data samples. (Setting range: 2 to 3000)	IN
lCorrelation	Correlation matrix of a unit space output enable/disable	Specify whether or not to output the correlation matrix of a unit space. • 0: Not to output the correlation matrix of a unit space. • 1: Output the correlation matrix of a unit space	IN
lSN	SN ratio output enable/disable	Specify whether or not to output the SN ratio (larger-is-better characteristic) of normal data MD. • 0: Not to output the SN ratio (larger-is-better characteristic) of normal data MD. • 1: Output the SN ratio (larger-is-better characteristic) of normal data MD.	IN
pMTUnitMDData	Unit space storage destination	Specify a structure to store data required for calculating a Mahalanobis distance.	IN/OUT
pMTUnitOutput	Output result storage destination	Specify a structure for storing output results.	IN/OUT

- The details of the MTUNIT_MDDATA_STRUCT structure are as follows.
  The memory to store output results should be allocated in a user-created program.

No.	Type	Argument	Name	Description	IN/OUT
1	float*	pfItemAverage	Average value output result storage destination	Specify the start address to store average values (array) for each item.	OUT
2	long	lItemAverageSize	Average value output result storage destination size (byte)	Specify the size of an array to store average values for each item in byte. For the array size, specify a value greater than or equal to the one specified to the number of normal data items (lItemNum) × number of bytes required for float type.	IN
3	float*	pfItemStandardDeviation	Standard deviation output result storage destination	Specify the start address to store the standard deviation (array) for each item.	OUT
4	long	lItemStandardDeviationSize	Standard deviation output result storage destination size (byte)	Specify the size of an array to store standard deviations for each item in byte. For the array size, specify a value greater than or equal to the one specified to the number of normal data items (lItemNum) × number of bytes required for float type.	IN
5	float*	pfInverseMatrix	Inverse matrix output result storage destination	Specify the start address to store the inverse matrix of the correlation matrix of a unit space (array).	OUT
6	long	lInverseMatrixSize	Inverse matrix output result storage destination size (byte)	Specify the size of an array to store the inverse matrix of the correlation matrix of a unit space in byte. For the array size, specify a value greater than or equal to the square of the value specified to the number of normal data items (lItemNum^2) × number of bytes required for float type.	IN

• The details of the MTUNIT_OUTPUT_STRUCT structure are as follows.
The memory to store output results should be allocated in a user-created program.

No.	Type	Argument	Name	Description	IN/OUT
1	float*	pfMahalanobisDistance	Normal data MD output result storage destination	Specify the start address to store the Mahalanobis distance (array) of normal data.	OUT
2	long	lMahalanobisDistanceSize	Normal data MD output result storage destination size (byte)	Specify the size of an array to store the Mahalanobis distance of normal data in byte. For the array size, specify a value greater than or equal to the one specified to the number of normal data samples (lSampleNum) × number of bytes required for float type.	IN
3	float*	pfCorrelation	Correlation matrix output result storage destination	Specify the start address to store the correlation matrix (array) of a unit space. The correlation matrix of a unit space is output when the correlation matrix of a unit space output enable/disable (lCorrelation) is '1'. (It is not output when the value is '0'.)	OUT
4	long	lCorrelationSize	Correlation matrix output result storage destination size (byte)	Specify the size of an array to store the correlation matrix of a unit space in byte. For the array size, specify a value greater than or equal to the square of the value specified to the number of normal data items (lItemNum^2) × number of bytes required for float type.	IN
5	float	fAverageMahalanobisDistance	Average value of normal data MD	The average value of normal data MD (1-point) is stored.	OUT
6	float	fLgeRespSN	SN ratio (larger-is-better characteristic) of normal data MD output result storage destination	The SN ratio (larger-is-better characteristic) (1-point) of normal data MD is stored. When 'SN ratio output enable/disable (lSN)' is '1', an SN ratio (larger-is-better characteristic) of normal data MD is output. (It is not output when the value is '0'.)	OUT

Description

- A unit space [*1] that is used in the MT method is determined based on the specified normal data.

*1 In a unit space, a correlation matrix, which is determined based on normal data, and its inverse matrix are included.

- Besides determination of a unit space, the following data can also be obtained with the DANL_MTUnit function.
  Average values for each item
  Standard deviations for each item
  Mahalanobis distances for each sample
  Average value of a Mahalanobis distance
  SN ratio (larger-is-better characteristic)
- Normal data is read from the input data storage destination (pfData). For the input data storage destination (pfData), data for the same number of units of data specified for the number of normal data items (lItemNum) × number of normal data samples (lSampleNum) should be stored in advance.

Ex.

Number of normal data samples (lSampleNum): 100 samples, number of normal data items (lItemNum): 10 items

lItemNum

lSampleNum	(1)	1	2	3	~	10
	1	pfData[0]	pfData[1]	pfData[2]	~	pfData[9]
	2	pfData[10]	pfData[11]	pfData[12]	~	pfData[19]
	~	~	~	~	~	~
	100	pfData[990]	pfData[991]	pfData[992]	~	pfData[999]

(1) Number of data points

- When 'SN ratio output enable/disable (ISN)' is '1', the calculation result of the SN ratio (larger-is-better characteristic) is output to the SN ratio (larger-is-better characteristic) of normal data MD storage destination (fLgeRespSN). The larger SN ratio (larger-is-better characteristic) corresponds to the better performance characteristic.
- If some items are strongly related to one another, an inverse matrix may not be calculated correctly and this is called multicollinearity. If it occurs, the processing will be interrupted and an error will be returned.
- The calculation result of the average value of a Mahalanobis distance is stored to the average value of normal data MD (fAverageMahalanobisDistance). If the average value is not an approximate value of 1, a multicollinearity error may occur; however, it is not regarded as an error.
- When a unit space is determined successfully based on sufficient number of samples without having a multicollinearity error, the average value of normal data MD (fAverageMahalanobisDistance) will be '1'. If a Mahalanobis distance of signal data is calculated using an output result (inverse matrix of correlation matrix) other than '1', an accurate result may not be obtained.
- If all the values of a specific item are the same, the standard deviation of the item will be '0' and therefore a Mahalanobis distance cannot be obtained. In that case, the processing will be terminated and an error will be returned.
- Before the execution of an operation, this instruction checks input values so as not to cause an overflow during operation. However, an error may occur depending on the combination of the input values even though the check has been conducted. If an overflow occurs during operation, 'Inf' or 'NaN' will be stored in the operation result. In that case, review the input data.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 50 Statistical analysis function

Relevant function

- Page 41 DANL_MTMahalanobisDistance

DANL_MTMahalanobisDistance

This function calculates a Mahalanobis distance of the specified input data.

Format

short DANL_MTMahalanobisDistance (float* pfData, long lDataSize, long lItemNum, long lSampleNum, long lContributionLevel, long lSN, MTUNIT_MDDATA_STRUCT* pMTUnitMDData, MTMD_STRUCT* pMTMD)

Argument

Argument	Name	Description	IN/OUT
pfData	Input data storage destination	Specify the start address of an array to be determined.	IN
lDataSize	Input data storage destination size (byte)	Specify a size of the array to be determined. Specify a value greater than or equal to the one specified to the number of input data items (lItemNum) × number of input data samples (lSampleNum) × number of bytes required for float type.	IN
lItemNum	Number of input data items	Specify the number of input data items. Set the same value as the one set for the number of normal data items (lItemNum) in DANL_MTUnit. (Setting range: 2 to 300)	IN
lSampleNum	Number of input data samples	Specify the number of input data samples. (Setting range: 1 to 3000)	IN
lContribution Level	Contribution level output enable/ disable	Specify whether or not to output a contribution level. • 0: Not to output a contribution level. • 1: Output a contribution level.	IN
lSN	SN ratio output enable/disable	Specify whether or not to output an SN ratio (larger-is-better characteristic) of input data MD. • 0: Not to output an SN ratio (larger-is-better characteristic) of input data MD. • 1: Output an SN ratio (larger-is-better characteristic) of input data MD	IN
pMTUnitMDD ata	Unit space storage destination	Specify a structure where a unit space is stored.	IN
pMTMD	Mahalanobis distance storage destination	Specify a structure to which a Mahalanobis distance is to be output.	IN/OUT

• The details of the MTUNIT_MDDATA_STRUCT structure are as follows.

No.	Type	Argument	Name	Description	IN/OUT
1	float*	pfItemAverage	Start address of average value storage destination	Specify the start address of an array where the average value of normal data is stored. For the array, the average value of items obtained by the DANL_MTUnit function should be stored.	IN
2	long	lItemAverageSize	Average value storage destination size (byte)	Specify a size of the array where the average value of normal data is stored. For the array size, specify a value greater than or equal to the one specified for the number of input data items (lItemNum) × number of bytes required for float type.	IN
3	float*	pfItemStandardDeviation	Start address of standard deviation storage destination	Specify the start address of an array where the standard deviation of normal data is stored. For the array, a standard deviation value obtained by the DANL_MTUnit function should be stored.	IN
4	long	lItemStandardDeviationSize	Standard deviation storage destination size (byte)	Specify a size of the array where the standard deviation of normal data is stored. For the array size, specify a value greater than or equal to the one specified for the number of input data items (lItemNum) × number of bytes required for float type.	IN
5	float*	pfInverseMatrix	Start address of inverse matrix storage destination	Specify the start address of an array where the inverse matrix of the correlation matrix of a unit space are stored. For the array, an inverse matrix value obtained by the DANL_MTUnit function should be stored.	IN
6	long	lInverseMatrixSize	Inverse matrix storage destination size (byte)	Specify a size of the array where the inverse matrix of the correlation matrix of a unit space is stored. For the array size, specify a value greater than or equal to the square of the value specified for the number of input data items (lItemNum^2) × number of bytes required for float type.	IN

- The details of the MTMD_ STRUCT structure are as follows.
  The memory to store output results should be allocated in a user-created program.

No.	Type	Argument	Name	Description	IN/OUT
1	float*	pfMahalanobisDistance	Input data MD output result storage destination	Specify the start address to store the Mahalanobis distance (array) of input data.	OUT
2	long	lMahalanobisDistanceSize	Input data MD output result storage destination size (byte)	Specify the size of an array to store the Mahalanobis distance of input data in byte. For the array size, specify a value greater than or equal to the one specified to the number of input data samples (lSampleNum) × number of bytes required for float type.	IN
3	float*	pfContributionLevel	Contribution level output result storage destination	Specify the start address to store a contribution level (array). A contribution level is output only when the contribution level output enable/disable (lContributionLevel) is '1'. (It is not output when the value is '0'.).	OUT
4	long	lContributionLevelSize	Contribution level output result storage destination size (byte)	Specify the size of an array to store contribution levels in byte. For the array size, specify a value greater than or equal to the one specified to the number of input data items (lItemNum) × number of input data samples (lSampleNum) × number of bytes required for float type.	IN
5	float	fLgeRespSN	SN ratio (larger-is-better characteristic) of input data MD storage destination	The SN ratio (larger-is-better characteristic) of input data MD is stored. When 'SN ratio output enable/disable (lSN)' is '1', an SN ratio (larger-is-better characteristic) of input data MD is stored.	OUT

Description

- A Mahalanobis distance of the specified input data can be obtained.
- Mahalanobis distances of input data*1 for each sample are calculated.

*1 Input data indicates data to check the error level. The distance between the data and a unit space is calculated as a Mahalanobis distance.

- Besides calculation of a Mahalanobis distance, the following data can also be obtained with the DANL_MTMahalanobisDistance function.
  Contribution levels for each item
  SN ratio (larger-is-better characteristic)
- By comparing contribution levels of each item, an error cause can be assumed.
- When 'SN ratio output enable/disable (lSN)' is '1', the calculation result of the SN ratio (larger-is-better characteristic) is output to the SN ratio (larger-is-better characteristic) of input data MD storage destination (fLgeRespSN). The larger SN ratio (larger-is-better characteristic) corresponds to the better performance characteristic.
- Input data for calculating Mahalanobis distance is read from the input data storage destination (pfData). For the input data storage destination (pfData), data for the same number of units of data specified for the number of input data items (lItemNum) × number of input data samples (lSampleNum) should be stored in advance.

Ex.

Number of input data samples (lSampleNum): 100 samples, number of input data items (lItemNum): 10 items

lItemNum

lSampleNum	(1)	1	2	3	~	10
	1	pfData[0]	pfData[1]	pfData[2]	~	pfData[9]
	2	pfData[10]	pfData[11]	pfData[12]	~	pfData[19]
	~	~	~	~	~	~
	100	pfData[990]	pfData[991]	pfData[992]	~	pfData[999]

(1) Number of data points

- If the values of the inverse matrix of the correlation matrix of a unit space are all '0', the processing will be interrupted and an error will be returned.
- If '0' is included in the standard deviation of items, the processing will be interrupted and an error will be returned.
- Before the execution of an operation, this instruction checks input values so as not to cause an overflow during operation. However, an error may occur depending on the combination of the input values even though the check has been conducted. If an overflow occurs during operation, 'Inf' or 'NaN' will be stored in the operation result. In that case, review the input data.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 50 Statistical analysis function

Relevant function

- Page 37 DANL_MTUnit

DANL_MultipleRegression

This function calculates a coefficient, constant, and regression statistics for multiple regression analysis.

Format

short DANL_MultipleRegression (float* pfDataX, long lDataXSize, float* pfDataY, long lDataYSize, long lPoint, long lDataItem, long lConst, long lState, MULTIPLEREG_STRUCT* pMultipleReg)

Argument

Argument	Name	Description	IN/OUT
pfDataX	X coordinate storage destination	Specify the start address of an X coordinate array to be calculated. For example, when an X coordinate to be calculated is X_0, X_1, ..., X_{n-1}, specify the start address of an X_0 coordinate array.	IN
lDataXSize	X coordinate storage destination size (byte)	Specify the size of an X coordinate array to be calculated. Specify a value greater than or equal to the one specified to the data size (lPoint) × number of data items (lDataItem) × number of bytes required for float type.	IN
pfDataY	Y coordinate storage destination	Specify the start address of a Y coordinate array to be calculated.	IN
lDataYSize	Y coordinate storage destination size (byte)	Specify a size of the Y coordinate array to be calculated. Specify a value greater than or equal to the one specified to the data size (lPoint) × number of bytes required for float type.	IN
lPoint	Data size	Specify the number of array elements of an X_i(i=0, 1, ..., n-1) coordinate and a Y coordinate to be calculated. The number of array elements that can be specified for each condition is as follows. • Necessity for calculating a constant b (lConst) = 0: 2 to 3000 • Necessity for calculating a constant b (lConst) = 1: 3 to 3000 Specify a value as follows. Otherwise, a multiple regression analysis error (627) occurs. When the necessity for calculating a constant b (lConst) is '0', specify a value greater than or equal to the one specified to the number of data items (lDataItem) + 1. When the necessity for calculating a constant b (lConst) is '1', specify a value greater than or equal to the one specified to the number of data items (lDataItem) + 2.	IN
lDataItem	Number of data items	Specify the number of X coordinates to be calculated. (Setting range: 1 to 64) For example, when an X coordinate to calculate is X_0, X_1, ..., X_{n-1}, the number of data items is n.	IN
lConst	Necessity for calculating a constant b	Specify whether or not to calculate a constant b. • 0: A constant b is '0'. • 1: Calculate a constant b.	IN
lState	Necessity for calculating regression statistics	Specify whether or not to calculate regression statistics. • 0: Do not calculate regression statistics. • 1: Calculate regression statistics. ■Regression statistics • Coefficient m standard error • Standard error of constant b • Coefficient of determination • Standard error of y • F statistical value • Degree of freedom • Regression sum of squares • Residual sum of squares	IN
pMultipleReg	Multiple regression analysis operation result	Specify the start address of the MULTIPLEREG_STRUCT structure to store an operation result.	IN/OUT

- The details of the MULTIPLEREG_STRUCT structure are as follows.
  The memory to store output results should be allocated in a user-created program.

No.	Type	Argument	Name	Description	IN/OUT
1	float*	pfMCoef	Coefficient m storage destination	Specify the start address of a destination to store a coefficient m (array).	OUT
2	long	lMCoefSize	Coefficient m storage destination size (byte)	Specify the size of a destination for storing a coefficient m (array). For the array size, specify a value greater than or equal to the one specified to the number of data items (lDataItem) × number of bytes required for float type.	IN
3	float*	pfMStdErr	Coefficient m standard error storage destination	Specify the start address of a destination to store a coefficient m (array) standard error.	OUT
4	long	lMStdErrSize	Coefficient m standard error storage destination size (byte)	Specify the size of a destination for storing a coefficient m (array) standard error. For the array size, specify a value greater than or equal to the one specified to the number of data items (lDataItem) × number of bytes required for float type.	IN
5	float	fBConst	Constant b	An operation result of the constant b (1-point) is stored.	OUT
6	float	fBStdErr	Standard error of constant b	An operation result of the standard error of constant b (1-point) is stored.	OUT
7	float	fDetermCoef	Coefficient of determination	An operation result of the coefficient of determination (1-point) is stored.	OUT
8	float	fYEstStdErr	Standard error of y estimation	An operation result of the standard error of y estimation (1-point) is stored.	OUT
9	float	fFStats	F statistical value	An operation result of the F statistical value (1-point) is stored.	OUT
10	float	fDF	Degree of freedom	An operation result of the degree of freedom (1-point) is stored.	OUT
11	float	fSSreg	Regression sum of squares	An operation result of regression sum of squares (1-point) is stored.	OUT
12	float	fSSresid	Residual sum of squares	An operation result of residual sum of squares (1-point) is stored.	OUT

Description

- A multiple regression analysis is performed for a specified X coordinate and Y coordinate.
  In the multiple regression analysis, the following formula should be applicable for X coordinate data and Y coordinate data.
  $Y=b+m_0 \times X_0+m_1 \times X_1+...+m_{n-1} \times X_{n-1}$
  (b: constant, m_i: coefficient (i=0, 1, ..., n-1))
- Data for multiple regression analysis is read from the X coordinate storage destination (pfDataX) for the data size (lPoint) × number of data items (lDataItem). As for Y coordinate, data is read from the Y coordinate storage destination (pfDataY) for the data size (lPoint). Store data to the X coordinate storage destination (pfDataX) and Y coordinate storage destination (pfDataY) as follows.

Ex.
Data size (lPoint): 100 points, number of data items (lDataItem): 10 items

			lDataItem			
	(1)	Y	X_0	X_1	~	X_9
lPoint	1	pfDataY[0]	pfDataX[0]	pfDataX[1]	~	pfDataX[9]
	2	pfDataY[10]	pfDataX[10]	pfDataX[11]	~	pfDataX[19]
	~	~	~	~	~	~
	100	pfDataY[99]	pfDataX[990]	pfDataX[991]	~	pfDataX[999]

(1) Number of data points

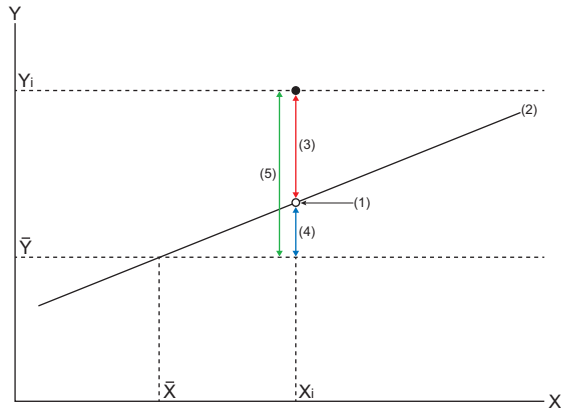
• An operation result is stored to the MULTIPLEREG_STRUCT structure.
The following results can be obtained in a multiple regression analysis.

No.	Argument	Name	Description
1	pfMCoef	Coefficient m storage destination	A regression coefficient m in a multiple regression formula is stored.
2	pfMStdErr	Coefficient m standard error storage destination	A standard error for the coefficient m is stored.
3	fBConst	Constant b	A constant term b in a multiple regression formula is stored.
4	fBStdErr	Standard error of constant b	A standard error of the constant b is stored.
5	fDetermCoef	Coefficient of determination	A coefficient of determination is stored. • Coefficient of determination = Regression sum of squares ÷ total sum of squares*1 (regression sum of squares + residual sum of squares) The range of the coefficient of determination is 0 to 1, and a higher coefficient is an indicator of a better goodness of fit for the observations. (A multiple regression formula obtained by multiple regression analysis will show the relationship between data items much more precisely.)
6	fYEstStdErr	Standard error of y estimation	A standard error of the predicted y value is stored.
7	fFStats	F statistical value	An F statistical value is stored. Determine if a measured relationship between an X coordinate and a Y coordinate is obtained by coincidence.
8	fDF	Degree of freedom	The number of units of measured data, which was independently collected, is stored.*2 • When the necessity for calculating a constant b (lConst) is '0': Data size (lPoint) - Number of data items (lDataItem) • When the necessity for calculating a constant b (lConst) is '1': Data size (lPoint) - Number of data items (lDataItem) - 1
9	fSSreg	Regression sum of squares	A sum of squares of residuals between an average value and a predicted value is stored.
10	fSSresid	Residual sum of squares	A sum of squares of residuals between a measured value and a predicted value is stored.

*1 For the regression analysis that passes through the origin, the total sum of squares is calculated as follows.

$$\sum(\text{Measured value } Y_i)^2$$

*2 In statistics, when there are n pieces of measured data that were independently collected, the group of data is referred to as 'degree of freedom n'.
The word 'independently' means that a precise value cannot be obtained from any measured value using other measured values.



$\overline{X}$, $\overline{Y}$: Average value
X_i, Y_i: Measured value
(1) Predicted value
(2) Regression line
(3) Remaining variation
(4) Regression variation
(5) Total variation

• If an overflow occurs during operation, 'Inf' or 'NaN' is stored in the operation result. Review the input data.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 50 Statistical analysis function

4 ERROR CODE LIST

This chapter shows error descriptions and corrective actions for error codes.

4.1 Data analysis functions

The following table shows the codes of errors which occur in data analysis functions and their corrective actions.

Error code		Description	Corrective action	Relevant function
Decimal	Hexadecimal			
-264	FEF8H	■Pointer error The address of the specified pointer is incorrect.	Check the address of the specified pointer.	☞ Page 13 DANL_SetOpCondition ☞ Page 14 DANL_GetOpCondition ☞ Page 15 DANL_DigitalFilter ☞ Page 18 DANL_EnvelopeCalculation ☞ Page 19 DANL_FFTSpectrum ☞ Page 21 DANL_FindCrossPoint ☞ Page 25 DANL_Peak ☞ Page 26 DANL_RMS ☞ Page 27 DANL_BoundCompareTest ☞ Page 29 DANL_AryBoundCompareTest
77	004DH	■Memory reservation error ■Resource shortage error ■Task over error Memory could not be reserved, or there are too many tasks using the following functions. • C Controller module dedicated function • C intelligent function module dedicated function • MELSEC data link function • MELSEC iQ-R series data link function • Data analysis function	• The memory may be insufficient. End another running task or reduce the access size. • Check if the C Controller module or the C intelligent function module is running normally. • Reset the C Controller module or the C intelligent function module. • Reduce the number of tasks using the target function, and retry. • Review the size or number specified to the arguments of the user program.	☞ Page 15 DANL_DigitalFilter ☞ Page 18 DANL_EnvelopeCalculation ☞ Page 19 DANL_FFTSpectrum ☞ Page 21 DANL_FindCrossPoint ☞ Page 27 DANL_BoundCompareTest ☞ Page 29 DANL_AryBoundCompareTest
256	0100H	■Data size (lPoint)/number of operating condition storage array elements (lSetNum)/number of operating condition storage array elements (lGetNum) out of range error The value of the specified data size or the number of operating condition storage array elements is out of the range.	Check if the value of the data size or the number of operating condition storage array elements is within the range.	☞ Page 13 DANL_SetOpCondition ☞ Page 14 DANL_GetOpCondition ☞ Page 15 DANL_DigitalFilter ☞ Page 18 DANL_EnvelopeCalculation ☞ Page 19 DANL_FFTSpectrum ☞ Page 21 DANL_FindCrossPoint ☞ Page 25 DANL_Peak ☞ Page 26 DANL_RMS ☞ Page 27 DANL_BoundCompareTest ☞ Page 29 DANL_AryBoundCompareTest
257	0101H	■Operating condition storage array (plSet) out of range error The value of the specified operating condition storage array is out of the range.	Check if the value of the operating condition storage array is within the range.	☞ Page 13 DANL_SetOpCondition
300	012CH	■Window type (lWindowType) out of range error The value of the specified window type is out of the range.	Check if the value outside the settable range is stored in the window type.	☞ Page 19 DANL_FFTSpectrum
301	012DH	■Spectrum format (lSpectrumFormat) out of range error The value of the specified spectrum format is out of the range.	Check if a value outside the settable range is stored in the spectrum format.	☞ Page 19 DANL_FFTSpectrum
302	012EH	■FFT operation error Data not available for FFT operation is specified.	Check if data available for FFT operation is specified.	☞ Page 19 DANL_FFTSpectrum
303	012FH	■Sampling cycle (lSamplingCycle) out of range error The value of the specified sampling cycle is out of the range.	Check if a value within the settable range is stored in the sampling cycle.	☞ Page 15 DANL_DigitalFilter

Error code		Description	Corrective action	Relevant function
Decimal	Hexadecimal			
304	0130H	■Frequency response filter type (lFilterType) out of range error The value of the specified frequency response filter type is out of the range.	Check if a value within the settable range is stored in the frequency response filter type.	☞ Page 15 DANL_DigitalFilter
305	0131H	■Cutoff frequency 1 (fCutoffFreqHz1) out of range error The value of the specified cutoff frequency 1 is out of the range.	Check if a value within the settable range is stored in the cutoff frequency 1.	☞ Page 15 DANL_DigitalFilter
306	0132H	■Cutoff frequency 2 (fCutoffFreqHz2) out of range error The value of the specified cutoff frequency 2 is out of the range.	Check if a value within the settable range is stored in the cutoff frequency 2.	☞ Page 15 DANL_DigitalFilter
307	0133H	■Digital filter type (lFilterCalcType) out of range error The value of the specified digital filter type is out of the range.	Check if a value within the settable range is stored in the digital filter type.	☞ Page 15 DANL_DigitalFilter
308	0134H	■Degree (lDegree) out of range error The value of the specified degree is out of the range.	Check if a value within the settable range is stored in the degree.	☞ Page 15 DANL_DigitalFilter
309	0135H	■Ripple (fRipple) out of range error The value of the specified ripple is out of the range.	Check if a value within the settable range is stored in the ripple.	☞ Page 15 DANL_DigitalFilter
310	0136H	■Digital filter operation error The value of the specified degree is out of the range.	Check if a value of the degree is smaller than that of the data size.	☞ Page 15 DANL_DigitalFilter
311	0137H	■Cross point recognition pattern (lCrossPattern) out of range error The value of the specified cross point recognition pattern is out of the range.	Check if a value within the settable range is stored in the cross point recognition pattern.	☞ Page 21 DANL_FindCrossPoint
312	0138H	■Maximum number of cross points (lMaxPoint) out of range error The value of the specified maximum number of cross points is out of the range.	Check if a value within the settable range is stored in the maximum number of cross points.	☞ Page 21 DANL_FindCrossPoint
313	0139H	■Check value specification error The specified check value is out of the range.	• Check if a check value is the upper limit value ≥ the lower limit value. • Check if a check value is the upper limit wave ≥ the lower limit wave.	☞ Page 27 DANL_BoundCompareTest ☞ Page 29 DANL_AryBoundCompareTest

4.2 Statistical analysis function

The following table shows the codes of errors which occur in statistical analysis functions and their corrective actions.

Error code		Description	Corrective action	Relevant function
Decimal	Hexadecimal			
-264	FEF8H	■Pointer error The address of the specified pointer is incorrect.	Check the address of the specified pointer.	☞ Page 31 DANL_LeastSquare ☞ Page 33 DANL_MovingAverage ☞ Page 35 DANL_StandardDeviation ☞ Page 36 DANL_Variance ☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance ☞ Page 44 DANL_MultipleRegression
77	004DH	■Memory reservation error ■Resource shortage error ■Task over error Memory could not be reserved, or there are too many tasks using the following functions. • C Controller module dedicated function • C intelligent function module dedicated function • MELSEC data link function • MELSEC iQ-R series data link function • Statistical analysis function	• The memory may be insufficient. End another running task or reduce the access size. • Check if the C Controller module or the C intelligent function module is running normally. • Reset the C Controller module or the C intelligent function module. • Reduce the number of tasks using the target function, and retry. • Review the size or number specified to the arguments of the user program.	☞ Page 31 DANL_LeastSquare ☞ Page 33 DANL_MovingAverage ☞ Page 44 DANL_MultipleRegression ☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
256	0100H	■Data size (lPoint) out of range error The value of the specified data size is out of the range.	Check if a value of the data size is within the range.	☞ Page 31 DANL_LeastSquare ☞ Page 33 DANL_MovingAverage ☞ Page 35 DANL_StandardDeviation ☞ Page 36 DANL_Variance ☞ Page 44 DANL_MultipleRegression
308	0134H	■Degree (lDegree) out of range error The value of the specified degree is out of the range.	Check if a value within the settable range is stored in the degree.	☞ Page 31 DANL_LeastSquare
314	013AH	■Number of moving average points (lAveragePoint) out of range error The value of the specified number of moving average points is out of the range.	Check if a value within the settable range is stored in the number of moving average points.	☞ Page 33 DANL_MovingAverage
315	013BH	■Least-squares method operation error Any of the variance values of the specified input data is '0'.	• Check that any of the variance values (dispersion of input data) of input data (X coordinate array and Y coordinate array) is not '0'.*1 • An overflow may have occurred during operation. Review the input data.	☞ Page 31 DANL_LeastSquare
316	013CH	■lItemNum out of range error	Check if a value within the settable range is stored in lItemNum.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
317	013DH	■lSampleNum out of range error	Check if a value within the settable range is stored in lSampleNum.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
318	013EH	■lCorrelation out of range error	Check if a value within the settable range is stored in lCorrelation.	☞ Page 37 DANL_MTUnit
319	013FH	■lSN out of range error	Check if a value within the settable range is stored in lSN.	☞ Page 37 DANL_MTUnit
320	0140H	■Multicollinearity error	Unable to calculate due to a multicollinearity error. Delete the data which are strongly correlated each other.	☞ Page 37 DANL_MTUnit
320+n (321 to 620)	0140H +nH (0141H to 026CH)	■Standard deviation error	Unable to calculate because the standard deviation of the item n (n=1, 2, 3) is '0'. All contents of item n are the same value. Delete the contents of item n, review the data, then execute the function again.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
621	026DH	■lContributionLevel out of range error	Check if a value within the settable range is stored in lContributionLevel.	☞ Page 41 DANL_MTMahalanobisDistance
622	026EH	■lSN out of range error	Check if a value within the settable range is stored in lSN.	☞ Page 41 DANL_MTMahalanobisDistance

Error code		Description	Corrective action	Relevant function
Decimal	Hexadecimal			
623	026FH	■Inversed correlation matrix error	The values of the inverse matrix of the correlation matrix of a unit space are all '0'. Check the values of the inverse matrix of the correlation matrix of a unit space obtained by the DANL_MTUnit function. After the correction, execute the function again.	☞ Page 41 DANL_MTMahalanobisDistance
624	0270H	■IDataItem out of range error	Check if a value within the settable range is stored in IDataItem.	☞ Page 44 DANL_MultipleRegression
625	0271H	■IConst out of range error	Check if a value within the settable range is stored in IConst.	☞ Page 44 DANL_MultipleRegression
626	0272H	■IState out of range error	Check if a value within the settable range is stored in IState.	☞ Page 44 DANL_MultipleRegression
627	0273H	■Multiple regression analysis error	A zero division occurred. Check the input data because multicollinearity data may be used.	☞ Page 44 DANL_MultipleRegression
628	0274H	■Input data storage destination size error	Check if the size set to the input data storage destination size is correct.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
629	0275H	■Average result storage destination size error	Check if the size set to the average result storage destination size is correct.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
630	0276H	■Standard deviation result storage destination size error	Check if the size set to the standard deviation result storage destination size is correct.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
631	0277H	■Inverse matrix result storage destination size error	Check if the size set to the inverse matrix result storage destination size is correct.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
632	0278H	■MD output result storage destination size error	Check if the size set to the MD output result storage destination size is correct.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
633	0279H	■Correlation matrix output result storage destination size error	Check if the size set to the correlation matrix output result storage destination size is correct.	☞ Page 37 DANL_MTUnit
634	027AH	■Contribution level result storage destination size error	Check if the size set to the degree of contribution result storage destination size is correct.	☞ Page 41 DANL_MTMahalanobisDistance
635	027BH	■IDataXSize out of range error	Check if the size set to the X coordinate storage destination size is correct.	☞ Page 44 DANL_MultipleRegression
636	027CH	■IDataYSize out of range error	Check if the size set to the Y coordinate storage destination size is correct.	☞ Page 44 DANL_MultipleRegression
637	027DH	■IMCoefSize out of range error	Check if the size set to the coefficient m storage destination size is correct.	☞ Page 44 DANL_MultipleRegression
638	027EH	■IMStdErrSize out of range error	Check if the size set to the coefficient m standard error storage destination size is correct.	☞ Page 44 DANL_MultipleRegression
639	027FH	■Overflow error	An overflow occurred during an internal operation. Review the input data.	☞ Page 37 DANL_MTUnit ☞ Page 41 DANL_MTMahalanobisDistance
640	0280H	■Unit space determination error	Check if the value set for ISampleNum is greater than or equal to the one set for the IItemNum.	☞ Page 37 DANL_MTUnit

*1 For the calculation formula of variance values, refer to the following section.
☞ Page 36 DANL_Variance

INDEX

MEMO

FUNCTION INDEX

D

MEMO

REVISIONS

*The manual number is given on the bottom left of the back cover.

Revision date	*Manual number	Description
April 2017	SH(NA)-081756ENG-A	First edition
December 2017	SH(NA)-081756ENG-B	■Added or modified parts TERMS, Chapter 1, Section 1.1, Section 1.3, Section 1.4, Section 1.5, Section 2.1, Section 2.2, Section 3.1, Section 3.2, Section 4.1, Section 4.2
September 2018	SH(NA)-081756ENG-C	■Added or modified parts Section 2.2, Section 3.2, Section 4.2

Japanese manual number: SH-081755-C

WARRANTY

Please confirm the following product warranty details before using this product.

1. Gratis Warranty Term and Gratis Warranty Range

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place. Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

(1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.

(2) Even within the gratis warranty term, repairs shall be charged for in the following cases.

1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
2. Failure caused by unapproved modifications, etc., to the product by the user.
3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

2. Onerous repair term after discontinuation of production

(1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued. Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.

(2) Product supply (including repair parts) is not available after production is discontinued.

3. Overseas service

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

4. Exclusion of loss in opportunity and secondary loss from warranty liability

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation to:

(1) Damages caused by any cause found not to be the responsibility of Mitsubishi.

(2) Loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products.

(3) Special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products.

(4) Replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

5. Changes in product specifications

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

TRADEMARKS

VxWorks and Wind River are either registered trademarks or trademarks of Wind River Systems, Inc.
The company names, system names and product names mentioned in this manual are either registered trademarks or trademarks of their respective companies.
In some cases, trademark symbols such as '™' or '®' are not specified in this manual.

SH(NA)-081756ENG-C(1809)
MODEL:DATAANALYSIS-CPRG-E

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN