

NAG C Library, Mark 26.1
CLW32261EL - License Managed
Microsoft Windows, 32-bit, Intel C/C++ or Microsoft C/C++

ユーザーノート

内容

1. イントロダクション	1
2. 追加情報	1
3. 一般情報	2
3.1. ライブラリのリンク方法	4
3.1.1. コマンドウィンドウ	6
3.1.2. Microsoft Visual Studio	8
3.1.3. その他の環境	12
3.2. Example プログラム	13
3.3. データ型	15
3.4. メンテナンスレベル	15
4. ルーチン固有の情報	16
5. ドキュメント	20
6. サポート	22
7. コンタクト情報	22

1. イントロダクション

本ユーザーノートは、NAG C Library, Mark 26.1 – CLW32261EL（ライブラリ）のご利用方法（リンク方法）を説明します。

本ユーザーノートには、NAG Library Manual, Mark 26（ライブラリマニュアル）には含まれない製品毎の情報が含まれています。ライブラリマニュアルに「ユーザーノート参照」などと書かれている場合は、本ユーザーノートをご参照ください。

ライブラリルーチンのご利用に際しては、ライブラリマニュアル（「5. ドキュメント」参照）の以下のドキュメントをお読みください。

- (a) How to Use the NAG Library and its Documentation
- (b) Chapter Introduction
- (c) Function Document

2. 追加情報

本ライブラリの動作環境やご利用方法についての最新の情報は、以下のウェブページをご確認ください。

<http://www.nag.co.uk/doc/inun/cl26/w321el/supplementary.html>

3. 一般情報

本ライブラリは、Intel® Math Kernel Library for Windows (MKL) が提供する BLAS/LAPACK ルーチンを利用するライブラリ (スタティック版と DLL 版) と、NAG が提供する BLAS/LAPACK ルーチンを利用するライブラリ (スタティック版と DLL 版) を提供します。

本ライブラリは、MKL version 2017.0.3 を用いてテストされています。MKL version 2017.0.3 は本製品の一部として提供されます。MKL の詳細については Intel 社のウェブサイト <https://software.intel.com/intel-mkl> をご参照ください。

パフォーマンスの面からは、MKL を利用するバージョンの NAG ライブラリ `nagc_mkl_MT.lib`, `nagc_mkl_MD.lib`, `CLW32261E_mkl.lib`/`CLW32261E_mkl.dll` のご利用を推奨します。これらのライブラリは NAG が提供する BLAS/LAPACK ルーチンを含みません。

また、MKL を利用しないバージョンの NAG ライブラリ `nagc_nag_MT.lib`, `nagc_nag_MD.lib`, `CLW32261E_nag.lib`/`CLW32261E_nag.dll` が提供されます。これらのライブラリは NAG が提供する BLAS/LAPACK ルーチンを含んでいます。

NAG ライブラリのスタティック版をご利用の場合は、共にリンクされる Microsoft ランタイムライブラリに従って、NAG ライブラリを選択する必要があります。マルチスレッドスタティックランタイムライブラリと共にリンクする場合は、`nagc_mkl_MT.lib` または `nagc_nag_MT.lib` をご利用ください。または、マルチスレッド DLL ランタイムライブラリと共にリンクする場合は、`nagc_mkl_MD.lib` または `nagc_nag_MD.lib` をご利用ください。

NAG ライブラリの DLL 版をご利用の場合は、インポートライブラリ `CLW32261E_mkl.lib` または `CLW32261E_nag.lib` をリンクしてください。実行時には、対応する DLL ファイル `CLW32261E_mkl.dll` または `CLW32261E_nag.dll` の格納フォルダーのパスが環境変数 `PATH` に設定されている必要があります。詳細は「3.1.1. コマンドウィンドウ」をご参照ください。

NAG ライブラリはメモリリークが起きないように設計されています。メモリの解放は NAG ライブラリ自身によってか、もしくはユーザーが `NAG_FREE()` を呼び出すことによって行われます。しかしながら、NAG ライブラリが依存している他のライブラリ（コンパイラのランタイムライブラリなど）がメモリリークを起こすかもしれません。このため、NAG ライブラリをリンクしているプログラムに対して何らかのメモリトレースツールを使った際に、場合によってはメモリリークが検出されるかもしれません。リークするメモリの量はアプリケーションによって異なると思われますが、NAG ライブラリの呼び出し回数に比例して際限なく増加するものではありません。

本製品で提供される MKL version 2017.0.3 はマルチスレッド化されており、環境変数 `OMP_NUM_THREADS` が設定されていない場合、複数のプロセッサまたはマルチコアチップを持つシステムでは、計算速度の向上のためにマルチスレッドで計算を行います。もし、MKL に複数のプロセッサまたはコアを使わせたくない場合は、環境変数 `OMP_NUM_THREADS` に “1”（もしくは、必要なスレッド数）を設定してください。なお、X06 ルーチンは MKL のスレッドの振る舞いに影響を与えません。

また、MKL には、条件付きビット単位の再現性（Bit-wise Reproducibility (BWR)）オプションがあります。

ユーザーコードが一定の条件（<https://software.intel.com/en-us/node/528579> 参照）を満たしていれば、環境変数 `MKL_CBWR` を設定することにより BWR が有効になります。詳細は MKL のドキュメントをご参照ください。しかしながら、多くの NAG ルーチンはこれらの条件を満たしていません。従って、MKL を利用するバージョンの NAG ライブラリの全ルーチンに対して、異なる CPU アーキテクチャに渡り `MKL_CBWR` による BWR を保証することはできません。BWR に関するより一般的な情報は、“How to Use the NAG Library and its Documentation” ドキュメントの「3.9.1 Bit-wise Reproducibility (BWR)」をご参照ください。

3.1. ライブラリのリンク方法

本セクションでは、以下のデフォルトのインストールフォルダーに本ライブラリがインストールされていることを前提とします。

C:\Program Files\NAG\CL26\clw32261el

(64-bit マシンでは、C:\Program Files (x86)\NAG\CL26\clw32261el)

もし、このフォルダーが存在しない場合は、システム管理者（本ライブラリをインストールされた方）にお尋ねください。以降の説明ではこのフォルダーを `install_dir` として参照します。

また、スタートメニューの NAG C Library (CLW32261EL) に以下のライブラリコマンドプロンプトのショートカットが置かれていることを前提とします。

NAG CLW32261EL Command Prompt

もし、このショートカットが存在しない場合は、システム管理者（本ライブラリをインストールされた方）にお尋ねください。また、本ライブラリのインストール時に作成される他のショートカットも同じ場所に置かれていることを前提とします。

NAG DLL (CLW32261E_mkl.dll/CLW32261E_nag.dll) をご利用の場合は、実行時に NAG DLL にアクセスできるように `install_dir\bin` フォルダーにパスを通してください。また、適切な Intel ランタイムライブラリにパスが通っていない場合は、`install_dir\rtl\bin` フォルダーにパスを通してください。また、MKL を利用する NAG DLL (CLW32261E_mkl.dll) をご利用の場合は、`install_dir\mkl_ia32_2017.0.3\bin` フォルダーにパスを通してください。この時、`install_dir\mkl_ia32_2017.0.3\bin` は `install_dir\bin` の後ろに設定してください。これは BLAS/LAPACK ルーチンのいくつかは、ベンダーバージョンとの問題を避けるために、NAG バージョン (CLW32261E_mkl.dll に含まれる) を使用する必要があるからです。（「4. ルーチン固有の情報」参照）

NAG DLL へのアクセスをチェックするために、スタートメニューの NAG C Library (CLW32261EL) にある以下のショートカットから診断プログラム NAG_C_DLL_info.exe を実行してください。

Check NAG CLW32261EL DLL Accessibility

この診断プログラムの詳細については、インストールノートの「4.2.2. アクセスチェック」をご参照ください。

3.1.1. コマンドウィンドウ

本ライブラリをコマンドウィンドウからご利用になる場合は、環境変数の設定が必要です。（なお、インストール時に環境変数の自動設定を選択された場合は、必要な環境変数はシステム環境変数に既に設定されています。）スタートメニューの NAG C Library (CLW32261EL) にある以下のショートカットがご利用いただけます。

NAG CLW32261EL Command Prompt

このショートカットは、本ライブラリと本製品で提供される MKL に対して必要な環境変数 INCLUDE, LIB, PATH を正しく設定した上でコマンドプロンプトを開きます。また、バッチファイル nagc_example_*.bat が必要とする環境変数 NAG_CLW32261EL も設定します。このショートカットを利用しない場合は、環境変数の設定を手動で行う必要があります。環境変数の設定はバッチファイル envvars.bat を用いて行うことができます。このバッチファイルのデフォルトの格納位置を以下に示します。

C:\Program Files\NAG\CL26\clw32261el\batch\envvars.bat

その後、以下に示すコマンドの一つでコンパイル／リンクを行ってください。
（ここで driver.c がユーザープログラムです。）

```
cl /MD driver.c CLW32261E_mkl.lib
```

```
cl /MD driver.c CLW32261E_nag.lib
```

```
cl /MT driver.c nagc_mkl_MT.lib mkl_rt.lib libiomp5md.lib user32.lib
```

```
cl /MT driver.c nagc_nag_MT.lib user32.lib
```

```
cl /MD driver.c nagc_mkl_MD.lib mkl_rt.lib libiomp5md.lib user32.lib
```

```
cl /MD driver.c nagc_nag_MD.lib user32.lib
```

注意：いくつかのコマンドは紙面の都合により二行で書かれていますが、実際は一行で打ち込んでください。

注意：ここでは Microsoft C コンパイラ `cl` を用いていますが、Intel C コンパイラ `icl` をご利用の場合は、上記コマンドの `cl` を `icl` に置き換えてください。どちらのコンパイラでもコンパイラオプションは同じです。

コンパイラオプション：

`/MD`

C ランタイムライブラリのマルチスレッド DLL バージョンのインポートライブラリとのリンクを指定するオプションです。

`/MT`

C ランタイムライブラリのスタティックマルチスレッドバージョンとのリンクを指定するオプションです。

`CLW32261E_mkl.lib` は MKL BLAS/LAPACK を利用する DLL インポートライブラリです。
`CLW32261E_nag.lib` は NAG BLAS/LAPACK を含む DLL インポートライブラリです。これらのライブラリは `/MD` オプションを付けてコンパイルされています。これらのライブラリを利用する場合には `/MD` オプションが必要です。

`nagc_mkl_MT.lib` は BLAS/LAPACK を含まないスタティックライブラリで、MKL スタティックライブラリとリンクする必要があります。`nagc_nag_MT.lib` は NAG BLAS/LAPACK を含むスタティックライブラリです。これらのライブラリは `/MT` オプションを付けてコンパイルされています。これらのライブラリを利用する場合には `/MT` オプションが必要です。

`nagc_mkl_MD.lib` は BLAS/LAPACK を含まないスタティックライブラリで、MKL スタティックライブラリとリンクする必要があります。`nagc_nag_MD.lib` は NAG BLAS/LAPACK を含むスタティックライブラリです。これらのライブラリは `/MD` オプションを付けてコンパイルされています。これらのライブラリを利用する場合には `/MD` オプションが必要です。

3.1.2. Microsoft Visual Studio

本セクションの説明は Microsoft Visual Studio 2015 を想定しています。
他のバージョンでは詳細が異なるかもしれません。

Visual Studio からの NAG ライブラリのご利用には、適切なオプション設定が必要です。

Visual Studio を起動して、通常通りにプロジェクトを作成してください。
以降の説明は、プロジェクトが開いていることを前提とします。

本ライブラリは最大最適化されています。そのため Debug モードだとランタイムライブラリについての警告メッセージが表示されますが、通常これは無視して構いません。
Release モードではこの警告メッセージは出力されません。Release モードへの変更は、ツールバーもしくはメニュー「ビルド > 構成マネージャー」から行うことができます。

本ライブラリは 32-bit ライブラリです。「構成マネージャー」の「プラットフォーム」が "Win32" に設定されていることを確認してください。

プロジェクトに NAG ライブラリを追加する手順を以下に示します。

1. プロジェクトのプロパティページを開いてください。

プロパティページは次のいずれかの操作で開くことができます。

- ソリューションエクスプローラーでプロジェクトを選択して、メニュー「プロジェクト > プロパティ」を選択してください。
- ソリューションエクスプローラーでプロジェクトを右クリックして、「プロパティ」を選択してください。
- ソリューションエクスプローラーでプロジェクトを選択して、ツールバーの「プロパティ ウィンドウ」ボタンを選択してください。
「プロパティ」ウィンドウの「プロパティ ページ」アイコンを選択してください。

2. 左パネルの「構成プロパティ > VC++ ディレクトリ」を選択してください.

- 「インクルード ディレクトリ」を選択して、
include フォルダーを追加してください.
- 「ライブラリ ディレクトリ」を選択して、
lib フォルダーと `rtl¥lib` フォルダーを追加してください.
(また、必要に応じて `mkl_ia32_2017.0.3¥lib` フォルダーを追加してください.)

各フォルダーのデフォルトを以下に示します.

インクルード ディレクトリ :

`C:¥Program Files¥NAG¥CL26¥clw32261el¥include`

ライブラリ ディレクトリ :

`C:¥Program Files¥NAG¥CL26¥clw32261el¥lib`

`C:¥Program Files¥NAG¥CL26¥clw32261el¥rtl¥lib`

`C:¥Program Files¥NAG¥CL26¥clw32261el¥mkl_ia32_2017.0.3¥lib`

変更を有効にするために「適用」ボタンをクリックしてください.

3. NAG ライブラリと Intel ランタイムライブラリ (また、必要に応じて MKL ライブラリ) をリンカオプションに指定します. 左パネルの「構成プロパティ > リンカー > 入力」を選択してください. 「追加の依存ファイル」に適切なライブラリファイルを追加してください. 【以下の表を参照】

また、リンカーエラーを回避するために、リンクオプションに `/SAFESEH:NO` を追加する必要があります. 左パネルの「構成プロパティ > リンカー > コマンドライン」を選択してください. 「追加のオプション」に `/SAFESEH:NO` を追加してください.

変更を有効にするために OK ボタンをクリックしてください.

4. 適切な C ランタイムライブラリのオプションを設定する必要があります。まず、C ソースファイル（例えば、NAG ライブラリの Example プログラムなど）をメニュー「プロジェクト > 既存項目の追加」からプロジェクトに追加してください（C ソースファイルがプロジェクトに無いと、C++ オプションが表示されません）。
- 再度、プロパティページを開いてください。左パネルの「構成プロパティ > C/C++ > コード生成」を選択してください。「ランタイム ライブラリ」を選択して、適切なバージョン（ご利用の NAG ライブラリに応じて、/MD または /MT）に変更してください。【以下の表を参照】

変更を有効にするために OK ボタンをクリックしてください。

NAG ライブラリ	MKL その他ライブラリ	C ランタイムライブラリ
CLW32261E_mkl.lib	（必要なし）	マルチスレッド DLL (/MD)
CLW32261E_nag.lib	（必要なし）	マルチスレッド DLL (/MD)
nagc_mkl_MT.lib	mkl_rt.lib libiomp5md.lib user32.lib	マルチスレッド (/MT)
nagc_nag_MT.lib	user32.lib	マルチスレッド (/MT)
nagc_mkl_MD.lib	mkl_rt.lib libiomp5md.lib user32.lib	マルチスレッド DLL (/MD)
nagc_nag_MD.lib	user32.lib	マルチスレッド DLL (/MD)

以上で、プロジェクトのビルド（コンパイル／リンク）を行うことができます。

Microsoft Development Environment 上でのプログラムの実行は、「デバッグ」メニュー（例えば、「デバックなしで開始（Ctrl+F5）」など）から行うことができます。実行時には、環境変数 PATH が適切に設定されている必要があります（「3.1. ライブラリのリンク方法」参照）。

プログラムの実行に入出力ダイレクションが伴う場合は、プロパティページの「構成プロパティ > デバッグ」から「コマンド引数」に適切なコマンドを指定してください。例えば、

< input_file > output_file

アプリケーションの作業フォルダー以外で入出力を行う場合は、フルパスもしくは相対パスでファイルを指定する必要があります。作業フォルダーの設定は、プロパティページの「構成プロパティ > デバッグ」から「作業ディレクトリ」で行うことができます。

3.1.3. その他の環境

その他の環境からの本ライブラリのご利用については、以下の追加情報ページをご参照ください。

<http://www.nag.co.uk/doc/inun/cl26/w321el/supplementary.html>

3.2. Example プログラム

提供される Example 結果は、nagc_mkl_MD.lib (MKL BLAS/LAPACK ルーチンを利用する NAG スタティックライブラリ) を用いて、インストールノートの「2.2. 開発環境」に記載されている環境で生成されています。Example プログラムの実行結果は異なる環境下（例えば、異なる C コンパイラ、異なるコンパイラライブラリ、異なる BLAS または LAPACK ルーチンなど）で若干異なる場合があります。そのような違いが顕著な計算結果としては、固有ベクトル（スカラー（多くの場合 -1）倍の違い）、反復回数や関数評価、残差（その他マシン精度と同じくらい小さい量）などがあげられます。

Example プログラムは本ライブラリが想定する動作環境に適した状態で提供されます。そのため、ライブラリマニュアルに記載／提供されている Example プログラムに比べて、その内容が若干異なる場合があります。

install_dir¥batch フォルダに 3 つのバッチファイル
nagc_example_DLL.bat, nagc_example_static_MT.bat, nagc_example_static_MD.bat
が提供されます。

これらのバッチファイルをご利用の際には、C/C++ コンパイラと NAG ライブラリに対して必要な環境変数が設定されていなければなりません。特に、環境変数 NAG_CLW32261EL に本ライブラリのインストール先（例えば、C:¥Program Files¥NAG¥CL26¥clw32261el）が設定されている必要があります。

これらのバッチファイルを用いて Example プログラムを簡単に利用することができます。これらのバッチファイルは、Example プログラムのソースファイル（必要に応じて、データファイル、オプションファイルその他）をカレントフォルダにコピーして、コンパイル／リンク／実行を行います。

ご利用の NAG ライブラリルーチンの名前をバッチの引数に指定してください。

例)

```
nagc_example_DLL e04ucc
```

この例では、e04ucce.c（ソースファイル）、e04ucce.d（データファイル）、e04ucce.opt（オプションファイル）をカレントフォルダにコピーして、コンパイル／リンク／実行を行い e04ucce.r（結果ファイル）を生成します。

- `nagc_example_DLL.bat`

`CLW32261E_nag.dll` (NAG BLAS/LAPACK を利用する NAG DLL ライブラリ)
をリンクします.

例)

`nagc_example_DLL e04ucc`

`CLW32261E_mkl.dll` (MKL BLAS/LAPACK を利用する NAG DLL ライブラリ)
をリンクする場合は `-mkl` オプションを付けてください.

例)

`nagc_example_DLL -mkl e04ucc`

- `nagc_example_static_MD.bat`

`nagc_nag_MD.lib` (NAG BLAS/LAPACK を利用する NAG スタティックライブラリ (/MD))
をリンクします.

例)

`nagc_example_static_MD e04ucc`

`nagc_mkl_MD.lib` (MKL BLAS/LAPACK を利用する NAG スタティックライブラリ (/MD))
をリンクする場合は `-mkl` オプションを付けてください.

例)

`nagc_example_static_MD -mkl e04ucc`

- `nagc_example_static_MT.bat`

`nagc_nag_MT.lib` (NAG BLAS/LAPACK を利用する NAG スタティックライブラリ (/MT))
をリンクします.

例)

`nagc_example_static_MT e04ucc`

`nagc_mkl_MT.lib` (MKL BLAS/LAPACK を利用する NAG スタティックライブラリ (/MT))
をリンクする場合は `-mkl` オプションを付けてください.

例)

`nagc_example_static_MT -mkl e04ucc`

3.3. データ型

NAG データ型 Integer と Pointer は、本ライブラリでは以下のように定義されています。

NAG 型	C 型	サイズ (バイト)
Integer	int	4
Pointer	void *	4

sizeof(Integer) と sizeof(Pointer) の値は a00aac の Example プログラムから得ることもできます。その他の NAG データ型の情報はライブラリマニュアル（「5. ドキュメント」参照）の “How to Use the NAG Library and its Documentation” ドキュメントをご参照ください。

3.4. メンテナンスレベル

ライブラリのメンテナンスレベルは、ライブラリルーチン a00aac の Example プログラムをコンパイル／リンク／実行することにより確認することができます。この時、バッチファイル nagc_example_*.bat を引数 a00aac と共に用いれば、Example プログラムのコンパイル／リンク／実行を容易に行うことができます（「3.2. Example プログラム」参照）。ライブラリルーチン a00aac はライブラリの詳細（タイトル、製品コード、使用されるコンパイラおよび精度、バージョン（Mark）など）を出力します。

または、診断プログラム NAG_C_DLL_info.exe を利用することもできます。

診断プログラムは其中で a00aac を呼び出します。

（インストールノートの「4.2.2. アクセスチェック」参照）

4. ルーチン固有の情報

本ライブラリルーチン固有の情報を（チャプター毎に）以下に示します.

a. f06, f07, f08, f16

nagc_mkl_MD.lib, nagc_mkl_MT.lib, CLW32261E_mkl.dll は, MKL の BLAS/LAPACK ルーチンを使用します. ただし, 以下の BLAS/LAPACK ルーチンは, MKL バージョンの使用に問題があるため, NAG バージョンが使用されます（呼び出されます）.

dgesvj dsbgvd zhbzgd zhegvx

b. s10 – s21

これらのチャプターの関数の動作は, ライブラリ実装毎に異なります.

一般的な詳細はライブラリマニュアルをご参照ください.

本ライブラリ固有の値を以下に示します.

s10aac $E_1 = 1.8715e+1$

s10abc $E_1 = 7.080e+2$

s10acc $E_1 = 7.080e+2$

s13aac $x_{hi} = 7.083e+2$

s13acc $x_{hi} = 1.0e+16$

s13adc $x_{hi} = 1.0e+17$

s14aac fail.code = NE_REAL_ARG_GT if $x > 1.70e+2$

fail.code = NE_REAL_ARG_LT if $x < -1.70e+2$

fail.code = NE_REAL_ARG_TOO_SMALL if $\text{abs}(x) < 2.23e-308$

s14abc fail.code = NE_REAL_ARG_GT if $x > x_{big} = 2.55e+305$

s15adc $x_{hi} = 2.65e+1$

s15aec $x_{hi} = 2.65e+1$

s15agc fail.code = NW_HI if $x \geq 2.53e+307$

fail.code = NW_REAL if $4.74e+7 \leq x < 2.53e+307$

```

fail.code = NW_NEG if  $x < -2.66e+1$ 

s17acc fail.code = NE_REAL_ARG_GT if  $x > 1.0e+16$ 
s17adc fail.code = NE_REAL_ARG_GT if  $x > 1.0e+16$ 
        fail.code = NE_REAL_ARG_TOO_SMALL if  $0 < x \leq 2.23e-308$ 
s17aec fail.code = NE_REAL_ARG_GT if  $\text{abs}(x) > 1.0e+16$ 
s17afc fail.code = NE_REAL_ARG_GT if  $\text{abs}(x) > 1.0e+16$ 
s17agc fail.code = NE_REAL_ARG_GT if  $x > 1.038e+2$ 
        fail.code = NE_REAL_ARG_LT if  $x < -5.7e+10$ 
s17ahc fail.code = NE_REAL_ARG_GT if  $x > 1.041e+2$ 
        fail.code = NE_REAL_ARG_LT if  $x < -5.7e+10$ 
s17ajc fail.code = NE_REAL_ARG_GT if  $x > 1.041e+2$ 
        fail.code = NE_REAL_ARG_LT if  $x < -1.9e+9$ 
s17akc fail.code = NE_REAL_ARG_GT if  $x > 1.041e+2$ 
        fail.code = NE_REAL_ARG_LT if  $x < -1.9e+9$ 
s17dcc fail.code = NE_OVERFLOW_LIKELY if  $\text{abs}(z) < 3.92223e-305$ 
        fail.code = NW_SOME_PRECISION_LOSS if  $\text{abs}(z)$  or  $\text{fnu}+n-1 > 3.27679e+4$ 
        fail.code = NE_TOTAL_PRECISION_LOSS if  $\text{abs}(z)$  or  $\text{fnu}+n-1 > 1.07374e+9$ 
s17dec fail.code = NE_OVERFLOW_LIKELY if  $\text{AIMAG}(z) > 7.00921e+2$ 
        fail.code = NW_SOME_PRECISION_LOSS if  $\text{abs}(z)$  or  $\text{fnu}+n-1 > 3.27679e+4$ 
        fail.code = NE_TOTAL_PRECISION_LOSS if  $\text{abs}(z)$  or  $\text{fnu}+n-1 > 1.07374e+9$ 
s17dgc fail.code = NW_SOME_PRECISION_LOSS if  $\text{abs}(z) > 1.02399e+3$ 
        fail.code = NE_TOTAL_PRECISION_LOSS if  $\text{abs}(z) > 1.04857e+6$ 
s17dhc fail.code = NW_SOME_PRECISION_LOSS if  $\text{abs}(z) > 1.02399e+3$ 
        fail.code = NE_TOTAL_PRECISION_LOSS if  $\text{abs}(z) > 1.04857e+6$ 
s17dlc fail.code = NE_OVERFLOW_LIKELY if  $\text{abs}(z) < 3.92223e-305$ 
        fail.code = NW_SOME_PRECISION_LOSS if  $\text{abs}(z)$  or  $\text{fnu}+n-1 > 3.27679e+4$ 
        fail.code = NE_TOTAL_PRECISION_LOSS if  $\text{abs}(z)$  or  $\text{fnu}+n-1 > 1.07374e+9$ 

s18adc fail.code = NE_REAL_ARG_TOO_SMALL if  $0 < x \leq 2.23e-308$ 
s18aec fail.code = NE_REAL_ARG_GT if  $\text{abs}(x) > 7.116e+2$ 
s18afc fail.code = NE_REAL_ARG_GT if  $\text{abs}(x) > 7.116e+2$ 
s18dcc fail.code = NE_OVERFLOW_LIKELY if  $\text{abs}(z) < 3.92223e-305$ 
        fail.code = NW_SOME_PRECISION_LOSS if  $\text{abs}(z)$  or  $\text{fnu}+n-1 > 3.27679e+4$ 
        fail.code = NE_TOTAL_PRECISION_LOSS if  $\text{abs}(z)$  or  $\text{fnu}+n-1 > 1.07374e+9$ 
s18dec fail.code = NE_OVERFLOW_LIKELY if  $\text{REAL}(z) > 7.00921e+2$ 

```

fail.code = NW_SOME_PRECISION_LOSS if $\text{abs}(z)$ or $\text{fnu}+n-1 > 3.27679\text{e}+4$
fail.code = NE_TOTAL_PRECISION_LOSS if $\text{abs}(z)$ or $\text{fnu}+n-1 > 1.07374\text{e}+9$

s19aac fail.code = NE_REAL_ARG_GT if $\text{abs}(x) \geq 5.04818\text{e}+1$
s19abc fail.code = NE_REAL_ARG_GT if $\text{abs}(x) \geq 5.04818\text{e}+1$
s19acc fail.code = NE_REAL_ARG_GT if $x > 9.9726\text{e}+2$
s19adc fail.code = NE_REAL_ARG_GT if $x > 9.9726\text{e}+2$

s21bcc fail.code = NE_REAL_ARG_LT if an argument $< 1.583\text{e}-205$
fail.code = NE_REAL_ARG_GE if an argument $\geq 3.765\text{e}+202$
s21bdc fail.code = NE_REAL_ARG_LT if an argument $< 2.813\text{e}-103$
fail.code = NE_REAL_ARG_GT if an argument $\geq 1.407\text{e}+102$

c. x01

以下の数学定数がヘッダーファイル nagx01.h に提供されます。

X01AAC (π) = 3.1415926535897932
X01ABC (γ) = 0.5772156649015328

d. x02

以下のマシン定数がヘッダーファイル nagx02.h に提供されます。

浮動小数点演算の基本的なパラメーター：

X02BHC = 2
X02BJC = 53
X02BKC = -1021
X02BLC = 1024

浮動小数点演算の派生的なパラメーター：

X02AJC = 1.11022302462516e-16
X02AKC = 2.22507385850721e-308
X02ALC = 1.79769313486231e+308

X02AMC = 2.22507385850721e-308

X02ANC = 2.22507385850721e-308

コンピューター環境のその他のパラメーター :

X02AHC = 1.42724769270596e+45

X02BBC = 2147483647

X02BEC = 15

5. ドキュメント

ライブラリマニュアルは本製品の一部として提供されます。

また、NAG のウェブサイトからダウンロードすることもできます。

ライブラリマニュアルの最新版は以下のウェブサイトをご参照ください。

<http://www.nag.co.uk/content/nag-c-library-manual>

ライブラリマニュアルは以下の形式で提供されます。

- HTML5 - HTML/MathML マニュアル（各ドキュメントの PDF 版へのリンクを含む）
- PDF - PDF マニュアル（PDF のしおり、または HTML 目次ファイルから閲覧する）

これらの形式に対して、以下の目次ファイルが提供されます。

nagdoc_cl26¥html¥frontmatter¥manconts.html

nagdoc_cl26¥pdf¥frontmatter¥manconts.pdf

nagdoc_cl26¥pdf¥frontmatter¥manconts.html

ライブラリマニュアルをインストールした場合、これらの目次ファイルはスタートメニューの NAG Mark 26-1 Manual にある以下のショートカットから開くことができます。

NAG C Library Manual Mark 26 (HTML5)

NAG C Library Manual Mark 26 (PDF)

NAG C Library Manual Mark 26 (PDF + HTML Index)

また、これらの目次ファイルへのリンクをまとめたマスター目次ファイルが提供されます。

nagdoc_cl26¥index.html

各形式の閲覧方法および操作方法については、以下のドキュメントをご参照ください。

http://www.nag.co.uk/numeric/cl/nagdoc_cl26/html/genint/essint.html#onlinedoc

加えて、以下のドキュメントが提供されます。

- in.html - インストールノート（英語版）
- un.html - ユーザーノート（英語版）

ユーザーノート（英語版）は、スタートメニューの NAG C Library (CLW32261EL) にある以下のショートカットから開くことができます。

NAG CLW32261EL Users' Note

6. サポート

製品のご利用に関してご質問等がございましたら、電子メールにて「日本 NAG ヘルプデスク」までお問い合わせください。その際、ご利用の製品の製品コード（CLW32261EL）並びに、お客様の User ID をご明記いただきますようお願い致します。
ご返答は平日 9：30～12:00, 13:00～17:30 に行わせていただきます。

日本 NAG ヘルプデスク

Email: naghelp@nag-j.co.jp

7. コンタクト情報

日本ニューメリカルアルゴリズムズグループ株式会社（日本 NAG）

〒104-0032

東京都中央区八丁堀 4-9-9 八丁堀フロンティアビル 2F

Email: sales@nag-j.co.jp

Tel: 03-5542-6311

Fax: 03-5542-6312

NAG のウェブサイトでは製品およびサービスに関する情報を定期的に更新しています。

<http://www.nag-j.co.jp/> （日本）

<http://www.nag.co.uk/> （英国本社）

<http://www.nag.com/> （米国）